



VOL:2 ISSUE:10 PRICE:£2.50

DIY TOOLKIT

MOUSE DRIVER INTO
MACHINE CODE

REVIEW:
QL-PC FILESERVER

VERY BASIC
SUPERBASIC
COLOUR CONTROL

TROUBLESHOOTER

PRINTING PUZZLES





Editor

Helen Armstrong

Publisher

Mark Kasprowicz

Advertising Manager

Jim Peskett

Creative Director

John Stanley

Graphic Artist

Steve Billington

Magazine Services

Linda Miller, Frances Maxwell,
Pauline Wakeling

**Sinclair QL World,
Published by Arcwind Ltd.
The Blue Barn,
Tew Lane, Wootton,
Woodstock,
Oxon. OX7 1HA
Tel: 0993 811181
Fax: 0993 811481
ISSN 026806X**

If you have any comments or difficulties please write to the editor and we will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies.

Back issues are available from the publisher price £2.50 UK, £2.99 Europe. Overseas rates on request.

Subscriptions from: Arcwind
The Blue Barn, Tew Lane,
Wootton, Woodstock, Oxon.
OX7 1HA
UK: £23.40
Europe: £32.90
Rest of World: £40.90

Reprographic Services:
Eclipse, Brook Street,
Watlington, Oxon. OX9 5JH.
Distributed by: Seymour Press
Ltd., Windsor House, 1270
London Road, Norbury,
London, SW16 4DH

© 1993 ARCWIND LTD.
Sinclair QL World is published 12 times a year. All rights reserved. Reproduction in whole or in part without written permission is strictly prohibited. We welcome contributions. All material must be supplied with a SAE.

While all reasonable care is taken in compiling QLWorld, the publisher and its agents assume no responsibility in affects arising therefrom. Opinions expressed are those of the authors.

Contents

VOL 2 ISSUE 10

4 TROUBLESHOOTER

Bryan Davies ponders on the precision (or otherwise) of print.

11 QL SCENE

Process controller from Di-Ren ... Programs for review.

12 OPEN CHANNEL

Postal safety ... Backnumbers ... stirrings in Sweden ... Clase Quill success ... Xchange query ... Psion 3 connections? ... Keyboard connection collapses.

14 QL SCENE

New Genealogy update ... All Formats Diary.

16 SUPERBASIC IN ACTION

Simon Goodwin diagnoses disk faults for all formats.

18 ALMOST A LAPWARMER

A personal view from Michael Jonas!

23 THE NEW USER GUIDE - part 28

The Keyword Index from TAN to WCOPY.

27 VERY BASIC SUPERBASIC

In Part 6, Dilwyn Jones goes into colour.

31 QL-PC FILE SERVER

Jack Brown tries a file swapping set-up from Di-Ren.

33 DIY TOOLKIT

Simon Goodwin's mouse drive goes into interrupt-driven QL machine code.

41 MACHINE CODE FOR BEGINNERS - PART 6

Alan Bridewell gets fit for practical applications.

43 REVIEW: ULTRAPRINT

Bryan Davies tests a high-spec screen dumper utility.

46 MICRO ADS

Coming Soon ...

Artificial Intelligence on the QL ... PDWorld Map conversion for the QL ... Italy show report and Canada show proposal ... Biorhythm program ...

TROUBLE SHOOTER

O dear! Bryan Davies loses ££ searching for prüf of a perfect character.

Text87 Plus-4 version 4 has been tested with the QXL card and works at screen resolutions up to 800 x 600 pixels. On the Atari, it has been tested at up to 1000 x 1000 pixels. The higher resolution allows more than one page to be displayed at a time. However, the effect is not yet as good as one would wish, because the image does not fill the screen.

Linedesign

Linedesign 2 is being tested, and should contain some welcome improvements. In particular, it redraws screen images a lot faster. This applies to both text as well, as this is displayed graphically too. In the past, text has usually been stored in "library" form (not by Linedesign, though) with each character pre-defined in shape and size. This has meant having individual files for each size, creating a mass of files which take up much disk space. The advantage of this method is that it is fast; you don't normally think about the characters being displayed on the screen when you type in Quill, because they appear more or less as you type.

Text in Linedesign 1 is a purely graphical affair, the founts being stored as sets of instructions for displaying the characters, rather than as pre-drawn characters. While this avoids masses of fount files, it results in slower text display, as each character is constructed, section by

section, on the screen. It is rather like a pavement artist, sketching a likeness of a passer-by. One option for Progs, the writers of Linedesign, is to switch to displaying text in bit-mapped form, using a few fixed sizes. Alternatively, they could simplify the design of screen founts, so that fewer calculations are needed to draw them. The same founts are (presumably) used for both screen display and printing, and any changes will affect what appears on the printed page. Programs on other computers often use different sets of founts for the screen and printer, to get around the problem of slow screen display. Printing is normally a secondary activity and users will put up with slowness there, especially if buffering is available to make printing a background job.

WYSIWYGuess

By now, many users will have realised that the WYSIWYG (what you see is what you get) we all wanted from wordprocessors is not quite what we hoped. Text87 gets around the display problem, to some extent, by providing a set of fixed screen founts. The user has to match these to printer founts as best possible; in some cases, they will look acceptable, but the small range of screen founts and sizes limits the WYSIWYG effect. Perfection approaches the problem in another way, representing different founts by variations

of the display colours, both foreground and background. The Founttext graphical driver for Text87 gives an actual WYSIWYG effect, except that the size of screen text differs from that of printed text. The characters on the screen are considerably larger than what appears on paper, making it necessary to choose sizes which are nominally much above what you want printed. One consequence of the larger display size is a limited amount of text visible on the screen. This may not be a great hardship for the user, and display is not noticeably slow. From the user's point of view, utilising exactly the same founts for screen and printer seems to be the way to go, and - by all accounts - the screen display on the obvious "micro" that does this (the NeXT) is impressive.

On other micros, the software writers have relied upon the hardware designers to get them out of the hole they have dug for themselves with WYSIWYG. Faster processors have covered up the slowness of current wordprocessors, but only for those fortunate enough to afford faster hardware. To a large extent, this has meant that wordprocessors run really well only on systems installed in companies where money is no object. The Gold Card has provided a considerable boost in speed, which has made some old programs - notably Quill - look a lot better than they used to do. However, we are favoured with two word processing

programs which are actually considerably faster, in most respects, than Quill. They achieve this by working in normal text mode, though, and it remains to be seen whether they will develop to display text that looks as printed, and whether they will then still be fast.

Founttext

The Founttext driver does give the desirable likeness between screen and paper, but it is limited by a fixed set of founts and sizes. There are no scalable founts (ones which can be used at almost any size). The founts supplied are not vector drawn ones and do show some jaggedness, even with a laser printer. The printer type supported is a 24-pin dot matrix; while some of these give surprisingly good print, it is not comparable with laser quality.

One big merit of Linedesign is that the range of sizes for text is virtually unlimited. In "real life" word processing, it is common for a range of sizes to be required for one fount - say, Helvetica 8-, 9-, 10-, 11- and 12-point. The bulk of the text may be in 10-point, with headings and sub-headings in 12- and 11-point. The smaller sizes are used for text which has lesser importance, such as footnotes, headers and footers.

Wet Ink

Ink-jet printers have brought the cost of good-

quality printing down. They can be messy, but are usually no worse than dot-matrix printers. Anyone who has not considered the cost of running ink-jet printers would do well to do so, though. They are not cheap to run. A rough calculation for the Canon BJ-10 BubbleJet suggests that replacement ink cartridges cost roughly 2-3 times the my laser print toner cartridges for the same number of printed sheets. The cartridges cost about the same, but seem to do a lot less sheets in the bubblejet.

One way of keeping cost down is to **refill** the ink cartridges. The procedure for doing this can be straightforward, provided you take a few precautions. The cost of ink refills is roughly half that of cartridges, but one supplier at least charges less (£6 plus VAT). There is always a risk when doing something like this during the guarantee period or a printer; the manufacturer and/or supplier might refuse to honour the warranty. This should not happen with ink-jet printers, because the cartridge is self-contained and the quality of the ink will normally not affect other parts of the mechanism. But do check your company literature carefully.

The first rule would seem to be not to let the cartridge nozzles dry up. There are 64 nozzles in the small print head of the bubblejet cartridge. Do not leave the printer unused for long periods, and refill cartridges as soon as they run out. If your printer has a self-clean function, use it often. You can clean cartridges out, but what is best to use I do not know. Isopropyl alcohol seems satisfactory, as it is a solvent but evaporates fairly quickly; it is used in the ink in original cartridges, so should not be harmful to them. As for washing with water - has anyone tried this often enough to know whether it is effective and safe?

If you have suitable ink, there is no obvious reason why you should not use colours, in both inkjet (that is, Hewlett-Packard DeskJet and similar) and bubblejet (Canon BJ and offspring). Roy Barber, former Quanta editor, uses ordinary Quink fountain pen ink and is satisfied with it. It does not dry out in as endorsing pad ink and others do. Advice from Alf Kendall is not to invest in bottles of "special" refill ink, not because the ink is not suitable, but simply because it may outlive your printer - the bubblejet cartridge contains roughly 10 millilitres. Work out how many refills you would need to finish a 1 litre bottle! Be sure to cover the filler hole after refilling, as you may tip the printer up at some later date with messy consequences.

Readers' Letters

Ken Attwood asks for advice on **saving** moderately long Quill documents. He has a document of 23 pages, but there are only 5000 words in it, and the lines are double-spaced. He does not say how large the file is, in terms of disk space, but it should take no more than 50-100 KB. The save operation fails with the message "I/O incomplete", and the original copy of the file is wiped from the disk.

One thing Ken does not say is what version of Quill he is using, and that is important. Earlier versions gave trouble saving longer documents, whereas the "final" version, 2.35, was quite good in this respect. In my own tests, it would work with files up to about 600 KB - far bigger than Ken's file and it fell over then for some reason other than the usual one (judging by the way it went).

So that is the first thought: if he is using 2.30 or earlier, he should get 2.35. Having said that, my recollection is

that 2.30 was really not bad, but anything prior to that is suspect (in more than one respect).

It apparently makes no difference what drive is being used - floppy, hard disk, or ram. His final comment is worth repeating: "I hope the problem can be overcome as I like Quill and prefer its simplicity... (to) Winword that I am... forced to use at work".

"Obvious?"

You cannot afford to ignore the "obvious" possibilities with such problems, because they are not always obvious to everybody. When Quill was written, memory was scarce; the QL had a total of 128 KB, of which 32 KB went for screen use straight away. Of the remaining 96 KB, Quill needs most, and you have maybe 20-30 KB for your work. Squeeze Quill by giving it only 60-70 KB to load into, and it is likely to express its disapproval by overflowing back onto disk. The famous file DEF.TMP is not created solely for holding the text of your document; it can be created simply by loading the program, if memory is short.

Assuming Quill is loading without overflowing, is the actual text file overflowing to disk? Ken has a Gold Card, so lack of memory should not be his problem. He may be using other programs at the same time; if they are loaded before Quill, there might be a shortage of memory by the time his document is loaded, but this seems unlikely. As people have found, when Quill is loaded first, nothing else gets a look in, because Quill grabs almost every byte of available memory. If his system has about 100 KB of free memory, at the time of loading his document, it should fit without overflow.

In the situation where document overflow occurs, disk space can be a factor. The temporary overflow file will

take up as much space as the saved document. That is, the maximum size of document you can work with, in the overflow situation, is about half the free space on the disk, so a 100 KB document needs something over 200 KB disk space. Should the free space on the disk drop near zero, and the on-screen document be increased in size, saving is clearly impossible.

One way of increasing the maximum file size you can work with is to load the document file from flp1; the temporary overflow file will go to flp2, allowing a theoretical maximum file size of near to the full capacity of a disk. This is hardly likely to help Ken, though, as his document is too small to need such measures.

Lost ...

Possibly the oldest "chestnut" in the micro printing world is the **missing £** sign. How many users have never had to figure out why the character printed bears no resemblance to the on-screen sign, or even why the screen character is different from the key pressed. To the computer novice, this is a big fiddle but, once you have spent a few years learning the tricks, it is easily dealt with. So why do we see the wrong character in magazines too? This comment applies mainly to amateur newsletters, but the glossies are not above it. Surely it's not all that difficult to print the UK currency symbol? (*This is the only sign people have trouble with over a period of years. It stems, or course, from the fact that UK and USA software and hardware arrange their currency signs differently, and if any one item in the chain - computer, software, printer, and any other complications like a daisywheel - is different, the £ sign will flip into something else. Incidentally, why does your computer never give mine a*

£ sign when everyone else's does?? Ed)

Another facet of the problem is the failure to print so-called "foreign" characters properly. One of our Norwegian readers has taken a bashing recently, with his name spelt wrongly in both **QL World** and **Quanta**. The problem is two-fold: the difficulty of transferring these accented letters between programs and computers, and possibly the lack of adequate proof reading just before the text goes to press. Maybe the errors are spotted but nothing is done about them. (We've replaced all your ^ signs with £ signs, Bryan. Count'em)

With the £ sign, we have a difficulty with the QL, because the internal character code is decimal 96, whereas the usual code is 156. The 156 is used in the IBM PC Extended Character Set, which is effectively the

standard on the PC. You may come across others, though; in the ANSI set, the £ is 163. The spreading monster Windows uses the ANSI set, creating two major groups of programs with differing character codes. A quick check through the various lists of codes handy on my shelf showed other possibilities - 11, 61 - although these come in special sets which replace the standard set, not extend it.

Making a Hash

The confusion is increased by the interchange of £ and #. The # is USA pound weight (eg 20#) or number (#1) symbol. Many people buy imported computers or keyboards, and these normally have the # on the 3 key. The code is 35. A UK keyboard has £ with the 3, so it is not surprising that the £ often gets

tied to the 35 code. The QL keyboard tries a "double-mix" - the # in the USA position, with the code 35, and £ on the key normally occupied by the # on UK PCs, but it has the code 96 rather than 156! You have to take the Character Set and Keys section of the QL User Guide with a pinch of salt when looking for codes, as there are several errors in some versions of it.

Returning to our Norwegian reader, Arvid Børretzen. His 'o' should have a slash through it. You can obtain a suitable display of the lower case form of this character by typing Ctrl-Shift-7, and a passable one from Ctrl-Shift-r; the second form has the slash rather too upright, however. But you are unlikely to get the same character from the printer, unless you use the Translate function (Quill) or equivalent, to send a code such as decimal 149 to the printer.

INFORMATION

Inkjet and bubblejet refill ink, £6 (black) and £8 (colour) plus VAT:
Concita Micro Systems Ltd.
Unit 12
Stirling Industrial Centre
Stirling Way
Borehamwood
Herts WD6 2BT.
Tel. 081 905 1707
Fax 081 207 5822

TF SERVICES

MINERVA

The ULTIMATE operating system upgrade

MKII MINERVA with battery for 256 bytes ram, CRASHPROOF clock & Philips PC bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

Other features common to MKI/MKII
DEBUGGED operating systems/ autoboot on reset/
Multiple Basic/ faster scheduler/graphics (10% of
lighting-string handling)/ WHEN ERROR/2nd screen/
TRACE/ foreign keyboard drivers/ "warm" fast reset/
Auto reboot after power failure. V1.97 now with built in
Multibasic and split OUTPUT baud rates with Hermes.

1st upgrade: free. Otherwise £3 (+ £5 for manual if req'd)
(send sac, Minerva & NEW disk/3 mds)
MKI to MKII upgrade - £30

MKI.....£40 MKII.....£65

GOLD CARD (v2.24+) COMPATIBLE

HERMES

A replacement QL co-processor for the QLs awful IPC 8049

- Do you get keyboard bounce?
- Do you find fast serial input unreliable?
- Do you want to connect a modem at 19200bps

If you can say YES to any of these, then you need HERMES

- 19200bps RELIABLE serial input - NO QCONNECT.
- Independent input baud rates - use serial mouse & print
- Stops keyboard bounce (unwanted repeat chrs)
- Improves "fuzzy" and "random" sound
- Provides extra input/output lines
- Key click

Fitting is simple. Remove the QL top (8 screws) & replace the chip marked 8049 or 8749 next to mdv 1.

£25 including manual/software

QL SPARES

Faulty QL board (no plug-in chips).....£9	Circuit diagrams.....£2
Keyboard membrane.....£9	1377 PAL.....£3
68008 cpu.....£8	Power supply.....£12
JM ROM.....£10	8301 ULA.....£10
8302 ULA.....£10	MDV ULA.....£12
8049 IPC.....£8	

Other components/sockets etc please phone

QL REPAIRS

Fixed price for unmodified QLs, excluding microdrives.
QLs tested with Thom-EMI ring and ROM software

£27 including 6 month guarantee

Prices include post & packing (UK only). Payment by Mastercard/Visa/Access/Burocard/cheque/postal order/PO Giro transfer (58 267 3909). MAIL ORDER ONLY - no calls without ringing first. Ring for overseas prices.



Holly Corner, Priory Road, ASCOT, Berks, SL5 8RL

Tel: 0344 890986 Fax & modem: 0344 890987



Telephone: 0753-888866
Fax: 0753-887149

(EEC)

W.N. Richardson & Co.

SINCLAIR QL PRICE LIST MARCH '93

18-21, Milsbourne House,
Chalfont St Giles, SL9 8UE

All Prices Include 17.5% VAT

QL Computers

JS COMPLETE

QL Computer, PSU, T.V. Lead, PSION V2.35 Software (Word Processor, Spreadsheet, Database & Business Graphics). Handbook for QL, Programs, & Superbasic. Free one years membership to QUANTA, the independent QL User Group with over 400 free library programs, Newsletters & H.E.P. 6 MONTHS WARRANTY.

JS £120

PART EXCHANGE £30 OFF ABOVE. Send in old QL (Unit only, any condition) - JS ROM £90 JM £79.00

BACKUP QLs QL & PSU only. JS £20 JM £65 (Part exchange allowance £15 if required)

Accessories

* NOTE: EXTERNAL (SER2) 3 BUTTON MOUSE AND SOFTWARE WITH EXTRA FUNCTIONS. NOW HERMES COMPATIBLE *

PC KEYBOARD INTERFACE, INTERNAL FITTING, FOR FITTING 102 KEY KEYBOARD	£ 75.00
PC KEYBOARD, UK version, 102 key (AT)	£ 30.00
PC KEYBOARD INTERFACE AND KEYBOARD (INCLUDES FREE JOYSTICK OR PSU)	£ 95.00
CASE AND LEAD FOR EXTERNAL FITTING of Keyboard Interface	£ 14.00
JOYSTICK with QL lead (no interface required)	£ 10.00
* QL MOUSE, 3 Button, Software controlled, externally mounted, simply fits in SER2	£ 45 CORDLESS MOUSE £ 55.00

Disk Drives

THE UNIVERSAL DRIVES ARE ALSO SUITABLE FOR ACORN BBC, ATARI ST, AMIGA, SPECTRUM, IBM, AMSTRAD AND OTHER PC COMPATIBLES

Universal 3.5" 1Mb disk drive, cased with PSU and free QL lead	£ 70.00	2 Units for dual drive	£120.00
Universal 3.5" 2Mb disk drive, as above but 2Mb capacity	£ 90.00	2 Units for dual drive	£170.00
3.5" 1Mb Uncased disk drive			£ 25.00
3.5" 2Mb Uncased disk drive			£ 49.00
Lead for uncased disk drives, or other universal micros	£10.00	POWER SUPPLY FOR UNCASSED DRIVES	£ 6.00
		METAL CASES FOR UNCASSED DRIVES	£ 6.00

Parallel Printers

NEW RANGE

SAMSUNG SP093 80 Col, 300 cps, 50 NLQ, 3K Print Buffer, Centronics, Tractor Feed/Sheet Feed, Paper Park
OLIVETTI COLOUR PRINTER, 24 pin, 200 cps, 50 NLQ, 40K Buffer, Epson LQ2350/IBM Compatible
CENTRONICS PARALLEL Printer Interface for above printers
QL LEAD FOR SERIAL PRINTERS £10

CANNON BJ10 EX

Monitors

PHILIPS 14" HIGH RES COLOUR EGA 31 DOT PITCH, FULL 85 COL WITH AMBER OR GREEN TEXT FEATURE, IDEAL FOR WORD PROCESSING, RECONDITIONED, 90 DAY WARRANTY.
OK FOR PC AND QL SELF SENSING AND MANY EXT'L CONTROLS.

£149
£39

Microdrive Cartridges and Spares

"MONO OPTION" WITH 9" GREEN SCREEN

4 New Cartridges in a wallet	£ 10.00	8 prog carts for reformatting in 2 wallets	£ 15.00
Plastic Storage filing box including 20 new cartridges	£ 3.00		£ 45.00
QL Psion Software V2.35 Includes Quill, Abacus, Archive, and Easel IN WALLET			£ 18.00
QL Psion Software Separate programs			£ 10.00
QL power Supply Unit	£ 10.00	Membrane (and instructions)	£ 9.00
TV or Network leads	£ 3.00	QL Top & Bottom Case	£ 5.00
ICs ZX £301 £ 6.00 ZX £302 £ 3.00		8049 (IPC) £ 3.00 MC 1377	£ 3.00
MC 68008 £ 12.00			
QL SERVICE MANUALS & CIRCUIT	£25.00		

S.A.E. FOR FURTHER DETAILS



Payment terms:
CWO, Access, VISA et cetera
Cheques - allow 10 days

Delivery:
Carriage - £9
Postage - £5



Product subject to availability, E&OE

TEL: 0753 888866

FAX: 0753 887149



New Files for Old Families!!

Chris Boutal's popular Genealogy program, **QL-Genealogist**, is now available for the pointer environment as **Genealogist-3**. The new version has new and better features as well as access to the PE.

When entering data, you can now choose from a "pick list" that pops up a list of your previous entries, allowing you to choose one to repeat instead of retyping it over and over again.

A new "Country" field gives each address a three-letter country code which allows you to search and report by country. You can define your own three-letter code if you wish.

Dates can be recorded as "approximate", "before", "after", "quarter" or "invalid" - the last means that if a date is recorded wrongly in an official course (such as a parish register) you can list both the wrong date and the correction for cross-referencing.

A new "Birth Brief" report gives a pedigree chart with birth, marriage and death dates and locations, extracted from the research data files.

Search facilities have been improved, with any string in a text field available for search, and a Pick List option in the Family Network. Search, load and save times for large Research Data segments have been improved. Gender (male, female or unknown) can be recorded and used in searches.

The old "Notes" feature has been replaced by two new facilities: the 'automatic' notes from the Research Data are now called "Events" and kept separately from your own notes. The new "Notes" is an extra research segment with place and source fields, as well as comment, event type and date.

Two windows in the Family Network can now be open at once, allowing you to keep track of where you are in your family more easily! And also to change relationship data using a mouse to point, rather than typing in reference numbers.

There are many other improvements, many suggested by active users of **QL-Genealogist Second Edition**. **Genealogist-3** comes with a complete new two-part manual and tutorial for new users and upgraders. The second part is an easy-lookup reference manual.

The new version requires a minimum 512K memory expansion. A conversion program is included to translate your old QLG files into the new format with new, faster loading times. Chris Boutal himself will translate your files for you if you send him a COPY of the data on a 3.5-in disk with formatted blank disks and return postage.

Prices for the programs are: **Genealogist-3** with necessary Pointer Environment files: £60; To upgrade from **QL-Genealogist Second Edition** £33; To upgrade from the original **QL-Genealogist First Edition** £45; to upgrade from the Budget (128K) **QLG** £50. All enquiries and orders to **Dilwyn Jones Computing, 41 Bro Emrys, Tal-y-Bont, Bangor, Gwynedd LL57 3YT, UK. Tel. 0248 354023.**

Hey! When will Dilwyn and Chris get together and offer **QLG** versions and **QL** hardware as a complete good-value family tree package? Family tracing is one of our most popular hobbies and there must be many family tracers who have never heard of the **QL**, who would jump at the chance of an all-in-one computer system at a bargain price.

QLScene

ALL FORMATS DIARY

Coming dates for the All Formats Computer Fair are:

24 Oct West: Brunel Centre, Templemeads Station, Bristol; **30 Oct** North east: Northumbria Centre, Washington Dist. 12; **31 Oct** Leeds University Sports Centre, Calverley St; **6 Nov** Oxford Cowley Parish Hall; **7 Nov** Brighton Corn Exchange, Church St; **13 Nov** West Midlands National Motorcycle Museum, M6 junction 23; **14 Nov** Cardiff University Union, Park Place; **20 Nov** London Sandown Park Racecourse **21 Nov** Portsmouth Guildhall **27 Nov** Haydock Park Racecourse, M6 J23 **28 Nov** Brunel Centre, Temple Meads Station, Bristol **4 Dec** Leicester De Montfort Hall, Granville Road **5 Dec** North East: Washington Leisure Centre, District 1.

The All Formats Fairs are going to many more venues now. Although you will not find **QL** traders at every one, they are good browsing places for multi-format users and general supplies. Check with suppliers whether they will be at a particular Fair. If you have far to travel phone All Formats 0608 663820 to check arrangements haven't changed.

Day tickets are £4; you can get up to 50 £1-off vouchers by sending an SAE to the organisers at: **Maple Leaf, Stretton-on-Fosse, Moreton-in Marsh, Gloucestershire GL56 9QX.** (Only one voucher per ticket.) Photocopies of these vouchers are OK. Admission is a flat £2 between 2pm and 4pm (£1-off vouchers do not apply at these times).

OPEN CHANNEL

Open Channel is where you have the opportunity to voice your opinions in Sinclair QL World. Whether you want to ask for help with a technical problem, provide somebody with an answer, or just sound off about something which bothers you, write to: Open Channel, QL World, The Blue Barn, Tew Lane, Wootton, Woodstock OX7 1HA.

Safer Post

Following last month's letter **Tragic Loss**, we decided to investigate postal protection further. What we discovered is this: currently a Certificate of Posting does provide some compensation for lost goods, but not the £150 as Mr. Fisher believed. The Post Office warned us that a rise in postal rates is due, and not to rely on current figures, but cover for normal post and therefore under a Certificate of Posting, as we write, is around £24.

For increased cover, you should consider using Registered Post or Registered Plus. Both these services cost in the region of £3.00 an item up to 2 kilograms weight. Registered post gives insurance cover of around £500, and Registered Plus around £1500 or £2000+, depending on the exact value of goods and charge. Additionally, both these services give a guaranteed next-day delivery, and signature on delivery. This gives owners extra security.

You should not risk sending items more valuable than the insurance cover given by these services, or any claim will be limited or invalidated. Parcels between 2 kg and 10 kg attract a higher charge, around £14. In this case it may be worth investigating Post Office Parcel Force, or another carrier.

Your repairer should always be willing to send your goods by protected post on request, even if he charges you for the service.

You can get leaflets on these services at your Post Office, but always check the date of printing (usually on the back page) in case prices out of date.

Back Issues

Thank you for your letter with information about the New User Guide. I note the information about sources of back numbers of QL World. Your help has been much appreciated.

E D Hyam
Seascale
Cumbria

In a nutshell: if Arcwind doesn't have the backissue you need, try Ron Dunnett at Qubbesoft. He has made a point of getting hold of some older ones.

Who and When?

Greetings.

I include a check to keep the subscription going for another 12 months. Now two little complaints: Why is no reference made in the entire magazine to the actual printing or issuing date? This is a very annoying source of confusion

when one tries to relate an ad or article to the actually concerned date!

Why don't you mention in your Club Access Sweden's Svenska QL Gruppen? When you mention International QL Conference which is among us of very reduced importance and according to its own SysOp's words "doesn't handle much about QL now-a-days"? Why not mention both instead? We very often mention you!

The complete address is Svenska QL Gruppen, Toftaasgatan 73, 421 74 Vastrå Frolunda, Sweden. I have NOT asked anybody's permission to pass on these informs, but I guess nobody will protest.

Attentiously yours,

Renato Costa Ferras
Spanga
Sweden

Thank you, Renato. There has clearly been a conference on the subject in Svenska in recent months, as Michael Cronsten of International QL Echomail Conferences sent us a very similar point of view about the Sweden clubs a few weeks ago and the information has been incorporated into Club. Club did not run in the last issue, but it should appear this time.

The clue to the issue date is found in the second digit of the issue number: 11.8 is the eighth issue for a given year, and relates to

the eighth month (August). Our grip on the calendar has been rather loose this year. We are trying to do something about it. Meanwhile - cheat: look at the All Formats Calendar and see which are the earliest dates we list. The issue should appear before those dates

Quill Driver

This letter refers to Howard Clase's printer driver in Psion Solutions, QL World June 1993. I used a routine like the one suggested for a major number of translates in some of my own SuperBasic programs, but had not thought of using it with Quill documents. Therefore "thank you" to Howard Clase.

Using my QLs, located in different places, with different printers (a Canon BJ10ex, an old Olivetti DM 290 and a still less recent Brother EP44), I dressed the routine (input from a .lis file, translates and output to printer) in a short program which starts by asking for the printer on duty and font required, reads the appropriate translate codes from DATA lines, and does the job. It is very simple.

The only trouble is (*But see below) that printing from Quill to a file, all type-face codes are lost (bold, underline, high, low). Is there any solution to this problem?

(*4 days later:) Please disregard the last para in my letter of October 10th. The trouble is not with the program, but with a copy of Psion Xchange which caused me other trouble too. Working with a different copy of Quill, everything is fine and the Quill codes for the typefaces are very evident in importing the .lis files from SuperBasic, and appear also to be the same as for my Olivetti printer.)

Ernesto P Braun
Rome
Italy

PSION 3

I am a sound engineer in the music business, and in 1986 I bought the original Psion Organiser II to keep my addresses and other data while I was on tour. This was the first computer I owned, and I quickly became addicted to playing with the programming language and writing a few programs. I started looking for a home computer for letter writing, and a friend who ran a rehearsal studio sold me a QL. This started a long relationship.

I made a database of all the concert venues I had been to, and transferred this, very sporadically and with many glitches, to my Psion. The only things I bought for the QL were the Ice rom and mouse, and a few Ice programs such as Ice and Mousart. With these and the old Psion Quartet I wrote all the stage specs and drew all the plans for Steve Harley and Cockney Rebel, Donovan, Roger Chapman, John Cale, Gary Glitter, etc.

Last year I moved to Germany and struggled with my girlfriend's IBM for a while before dragging out my old QL. I invested in a Gold Card and suddenly realised why I had never dropped the QL before. Since then I have bought

new disk drives, Xchange, and am struggling along with Qpac 2 since no-one is writing any more Ice programs. (A pity, since my Ice-revised Psion suite worked a treat, and having all the commands from Archive available as a mouse-driven menu made it so much easier to use. Why has no-one written a pointer program like that?).

Someone stole my briefcase in Paris last year and I lost my old Psion Organiser (but not the data, as I had it all on a spare eeprom at home) so I have now upgraded to the new Psion Series 3, a fine machine, but how do you connect it to the QL?

I fully intend to carry on using the QL, and to master Qpac and the pointer system (I have just ordered QDesign and the Minerva rom), and this is not only because I like the machine so much. When I was back in England recently, I visited Ron Dunnet of Qubbesoft, and David Johnson, both of whom gave me good advice and were very friendly. (When's the last time your IBM supplier gave you a cup of tea in his home and explained a few programs?) and Jochen Merz over here is just as open and helpful.

So here's to the QL community and long may it survive.

Roy Wood (Aural
Architect and Mixer
Manipulator)
Hamburg
Germany

PS Does anyone have an old copy of QLPaint for sale? Mine dies with the "bad or changed medium" message and the problem seems to be the master microdrive which you need to start the program. Also I am missing to issues of QL World, May 91

and April 92, so if anyone has any spares ... oh, and why does **Glossary in Xchange** not work?

More questions in the PS than the letter! OK: Does anyone have a Psion Organiser 3 connected to a QL. Come to think of it, someone at Psion might know. Might. And here's me telling everyone that Ron Dunnett has backissues. Well, he can't have them all. I'll look into it. And why does Glossary in Xchange not work?

Random Keys

My Keyboard Products (ed. note: Not seen in the QL business recently.) keyboard has gone on the blink, it types out random letters before typing the letter that you have just punched up.

Can anyone guesstimate

which chip is likely to be causing the problem?

I managed to get a keyboard from Bill Richardson, who went out of his way to help, but unfortunately the pin-out is different.

Who can tell me how to connect an Epson type 3 keyboard 5-pin DIN plug (type a) to a 6-pin-socket Keyboard Products interface?

As you can see from this typing, I am in urgent need of a working computer. Oh for a typewriter with a spellchecker!

John McNaught
Ralston
Paisley

PS - I am no slouch with a soldering iron.

Any information, please, to John via QL World. Bill was one person I would have mentioned. Try Tony Firshman too.

Editor's Notebook

Dilwyn Jones' SuperBasic for Beginners is back this month, as is Alan Bridewell's Beginners; Machine code. Dilwyn is a busy man at the moment so we can't promise another episode next month, but he'll be back before too long.

Well, it looks as though I was right about finishing this issue as you were reading the last one; although I expected both to be a few days earlier. Believe it or not, we started earlier this month, and expect to start earlier again next month until we bear more relation to Real Time.

Tony Firshman is in the process of launching some new and interesting pieces of small-scale interface hardware (see our Eindhoven Show report earlier this year for a preview), of which more news later.

The brightly coloured birdie on the cover, codenamed the Goodwin Budgie, is the collaborative product of Dave Barker's drawing, the QL's native colours, SuperBasic In Action's colour separation codes and HPDUMP from DIY Toolkit.

Process Controller from Di-Ren

Hardware and software makers Di-Ren have reintroduced their **Micro Process Controller** (MPC) computer-controlled switching unit after revisions. The self-contained process controller can switch AC and DC voltages up to a maximum of 3 amps at 240 volts AC (mains). The internal electromechanical relays are controlled by sending information from a computer program to the MPC, which controls up to six circuits. A second MPC can be connected in-line, allowing control of up to 12 circuits.

A single MPC can be connected in-line with a printer, allowing the use of a printer and the MPC simultaneously. The MPC can be run as a background task by multitasking machines. PCs can use a TSR program for the same function, so that the computer is not tied up by the MPC unless very fine timings are required.

Low-cost programs that allow both background and critical time control are available already for the QL and PCs from Di-Ren.

The user documentation contains, as well as programming and connection instructions, suggestions on how remote control via a modem might be achieved.

The MPC needs a power supply of between 9 to 12 volts DC, 300 milliamps. This can be provided by 9 volt batteries, but for steady use a battery eliminator (external power supply) is recommended.

The MPC is aimed at the hobbyist, educational and light industrial markets. Current users wishing to upgrade should contact **Di-Ren** on 0922 33580. For more information on current prices please contact **Dilwyn Jones** on 0248 354023.

There is a colour brochure available. Prices given suggest that a working MPC with battery eliminator could be set up for around £100.

More Programs to Review!

QL World has copies of some interesting programs which haven't been reviewed over the last year. We would particularly like to hear from anyone who has been using any of these programs, but if you have not and are interested in the subject, drop the Editor a line at address on the **Open Channel** pages.

From Jochen Merz in Germany we have two games, **Minefield** (which requires the Pointer Environment and Toolkit 2) and **The Oracle**, a game with a mysterious solution! Both are on disk.

From Dilwyn Jones Software, a suite of four programs by Care Electronics: **Locksmith** and **4Matter**, two data-copying programs, the former for microdrive only, the latter for microdrive and disk data; **MDV Toolchest**, a collection of SuperBasic extras concerned with reading and writing mdv sectors, monitoring, and sector editing. On mvd; needs knowledge of file handling and Qdos. **Sidewinder**, which we have already mentioned - a variable-size screen dump printer, including labels and banners. Disk only. Also a small screen blanker program, **Screen Dazzler**, by Bruce Nicholls, on disk.

From Digital Precision, we have a number of Archive-related utilities, several ex-PDQL. Some are recommended for "Archive techies" and some for "non-Techies." The "Non-techie" ones are: **Mailmerge**; **Names and Addresses Database System**; **Dat-A-Point Appointment Manager**; **Recover**; **Archive Tutorial**; and a separate 'odds and ends' batch: **Compare**, and **Copy**.

The "Techie" ones are: **Archdev+RTM**; **Sedit** and **Screenprint**; **Pedit**; **Database Monitor** and **Analyser**.

All these are on disk.

From SQLUG shareware, we have **Disktidy** with a PD index application (two disks).

We also have a batch of small, low-cost disk programs from QBits: **Conundrum** (a word game); **Early Learn** (spelling and time for young children); **Storeman Sam** (profit and loss educational game); **QL Engine Demo** (Attractive graphics demo. Needs Gold Card. Two disks). All these are on disk.

Please contact the Editor if you would like to review any of these programs.

SuperBasic In Action

Simon Goodwin checks out the latest Gold Card drives and explains fault diagnosis for all QL disk formats.

This SuperBasic program is very useful despite its shortness. It enables you to **test** your disks. If you find bad sectors you can recover the remaining data with **Revive**, the disk utility that launched SuperBasic in Action at the start of the year.

The original version of this program was one of the **DIY Toolkit** disk utilities, released three years ago. Since the Gold Card has come out, bringing high and extra-high density disks to the masses, I have updated the program to work with all three types of disk.

The new file has been added to **DIY Toolkit Volume D**, along with **Revive** and other utilities for the new disk formats. You can still use it with the oldest 180K and 360K drives as long as you change a couple of program lines, as noted later.

New Drives

The Gold Card gives much accelerated disk access, as well as faster processing. All the disk formats turn at the same speed, but higher densities use extra current and special coatings to record more on each track. The program can test Gold Card ED disks at around 34K per second, checking 3.2 megabytes in about the time they take to **FORMAT**.

The secret of this speed is

the **FOR** loop in line 530. This means that sectors are read in a scrambled order from each track; the scrambling or 'interleaving' gives the QL just enough time to process the last sector, and be ready for the next sector number in the sequence.

The original QL needed to let two sectors go by while it processed each one, but the faster Gold Card can read alternate sectors under SuperBasic control. The disk format of HD and ED disks uses this faster 1:2 interleave, and the **FOR** loop in **CHECK_DISK** works at that speed. Substitute the **FOR** on line 520 for the one on line 530 if you're using a QL or slow system.

The interleave makes no difference in the case of Amiga Qdos, as it reads a track at a time. This is one of the few floppy disk utilities that runs faster on Amiga Qdos than on a real QL, but the difference is small and more down to Mark J Swift's rewrite of the disk handler than the Amiga hardware.

The latest Amigas have high density drives, and Mark's brother Frank has started work on a driver for QL HD disks. You should be able to use **CHECK_DISK** with DD and HD disks on the PC once the QXL SuperBasic interpreter is working, but you may need to tweak the interleave to suit the rate your machine passes sectors between the PC and Qdos.

So far ED disks are solely

the domain of the Gold Card. Sector number ten is only read in line 500 if this is an ED disk. There is an extra delay of five sectors to give SuperBasic plenty of time to interpret its way to the next **GET** command. Some PC, ST and Spectrum formats use a tenth sector to squeeze more onto the disk, so you might like to change the **IF** test so that they can be checked on the QL.

The program starts by asking you the density of the disk you want to check: type E, H or D. The SuperBasic will not run on the old AH and JM systems because it uses **WHEN ERROR** to continue after bad tracks. It suits Minerva, JS, MG, Thors and Amiga Qdos, although all but Minerva have faults that can lock the keyboard if **WHEN ERROR** is invoked from inside an expression. **DISK_CHECK** is safe as it does not do that. Just in case of a typing error, remember to **SAVE** the program before running it for the first time.

Once the density is set the program runs automatically, testing each track and generating a high-resolution colour map of good and bad tracks. If you use the Qpac2 window manager the display will vanish as soon as the program finishes. I am told that Qpac users get time to read the summary of faults and time taken if they add:

```
660 PAUSE : HOT_DO "b"
```

at the end of the program, and press any key when they want to get back to SuperBasic. By default **CHECK_DISK** uses "FLP1_" as everyone has that drive, but you can direct it elsewhere by changing **DRIVE\$** in line 410.

Compilation

The program is compilable but its speed is limited by the drive and device-driver, not the processor, and you won't be able to do anything else with the drive while it is being checked. If compiling with Turbo, remove the second parameter 'sector' from the **GETs** at line 500 and 540, and add this after each **GET**:

```
sector$=INPUT$(#3,512+1  
536*(k$="--e"))
```

Use **SET_POSITION** instead of **GET**, and your task will not need Toolkit 2 and can be distributed with the Runtime Toolkit.

Old Drives

When QL disk drives were a rarity the 720K type was the most expensive. Many people started out with single-sided disks or drives with half the number of tracks: 40 instead of 80. The disk checker will work with these old disks as long as you tell it the number of tracks or sides.

Line 150, right at the start

of the program, sets the number of the last track. Tracks are numbered from zero, so this is 39 for a 40 track drive and defaults to 79, for 80 tracks. Regardless of the data density, all current 3.5 inch drives disks use 80 tracks, but place increasing amounts of data on each track: 4.5K per side for DD, 9K for HD and 20K for ED, which uses ten blocks of 2K rather than the usual 512 byte sectors.

Single sided drives have only one head to read and write the disk, halving their capacity. Sides are numbered zero and one, and single sided drives lack side 1. Change line 480 to:

```
480 FOR side=0
```

if you have a single sided drive. This only goes once round the loop, so you can replace both the FOR and END FOR (line 590) with a simple assignment:

```
445 side=0
```

This is more efficient as the assignment is not made for every track. It's hardly worth making this change if you have a mixture of drives as you need the FOR loop again to use a double-sided drive, but you might like to give the user the option to set the number of sides and tracks.

Diagnosis

If you find bad tracks on a disk you can usually deduce the cause from their distribution. Software faults usually clobber one or both sides of track zero. This holds the directory and map, which are altered more often than later tracks.

Mechanical damage, including coffee and Ribena, normally shows up as groups of bad tracks on one side. Some vendors format their disks to just the capacity needed for the files; these will show 'bad tracks' where there is no format.

QL World SuperBasic in Action DD/HD/ED disk checker

```
100 REMark QL/Thor XVI/Gold Card VIEW_DISK_GC
110 REMark Copyright 1992,93 Simon N Goodwin
120 REMark Optimised interleave for GOLD CARD
130 REMark scans ED disks at around 34K/sec.
140 REMark V 0.8, 18-10-93, Needs WHEN ERROR
150 LET max_track=79 :REMark 39 for 40 tracks
160 :
170 WHEN Error
180   IF ERLIN=500 OR ERLIN=540
190     PRINT #5;" Track ";true_track;" side ";side<>0;" BAD "
200     bad=bad+1 : eek=1 : INK 4,2 : CONTINUE
210   END IF
220   PRINT #0;"Error at ";ERLIN;:REPORT #0 : STOP
230 END WHEN
240 :
250 MODE 4 : OPEN #4,scr_512x256a0x0 : PAPER #4,0 : CLS #4
260 OPEN #4,"scr_330x244a180x0" : BORDER #4,1,7
270 BORDER #4,2,128 : PAPER #4,0 : INK #4, 7 : CLS #4
280 SCALE #4,170,-84,-85 : CIRCLE #4,0,0,84 : CIRCLE #4,0,0,4
290 AT #4,1,1 : PRINT #4;"Track ";
300 OPEN #5,"scr_168x240a8x2"
310 PAPER #5,0,2 : INK #5,7 : CLS #5 : CSIZE #5,1,0
320 PRINT #5;" VISUAL DISK CHECKER"\\
330 PRINT #5;"Pick capacity: D=720K"\\ "H=1.44 Meg E=3.2 Megs"\\
340 REPEAT kpoll
350   k$=INKEY$
360   IF k$=="d" OR k$=="h" OR k$=="e" : EXIT kpoll
370 END REPEAT kpoll
380 IF k$=="e" : dev$="*D4E" : ELSE dev$="*D2" & k$
390 STRIP #5,2
400 DIM drive$(8) : bad=0
410 drive$="flp1_" : sector$=FILL$(0,512+1536*(k$=="e"))
420 t=DATE
430 OPEN IN #3,drive$ & dev$
440 AT #4,1,44 : PRINT #4;drive$
450 FOR true_track=0 TO max_track
460   AT #4,1,7 : INK #4,7 : PRINT #4,true_track;
470   track=true_track*65536
480   FOR side=0,256
490     eek=0 : INK #4,5 + (side<>0)
500     IF k$=="e" : GET #3\track+side+10,sector$
510     IF NOT eek
520       REMark QL: FOR sector=3,7,2,6,1,5,9,4,8
530       FOR sector=6,8,1,3,5,7,9,2,4
540         GET #3\sector+side+track,sector$
550         IF eek : INK #4,2 : EXIT sector
560       END FOR sector
570     END IF
580     CIRCLE #4,0,0,83-true_track+side*2E-3
590   END FOR side
600 END FOR true_track
610 AT #4,22,1 : PRINT #4,bad;" fault";
620 IF bad<>1 : PRINT #4;"s";
630 AT #4,22,44 : PRINT #4;DATE-t;" s";
640 REMark QPAC users may need: PAUSE : HOT_DO 'b'
650 CLOSE #3 : CLOSE #4 : CLOSE #5
```


Manufacturing faults show up as disks which fail to format to their full capacity - perhaps 1386 sectors instead of 1440, indicating six unusable tracks. CHECK_DISK will tell you which tracks are bad; if they are all on side 1 you might like to play it safe and reformat the disk to use only the good side:

FORMAT "FLP1_SingleSide**"

The eleventh character of the name must be an asterisk. It is not recorded but tells the disk system to prepare a single-sided disk. The quotes are needed so that SuperBasic does not confuse the asterisk with a multiplication symbol.

Inner Limits

The innermost tracks of a disk have highest numbers, and the highest data-density due to their reduced circumference. If only the inner tracks are bad, it might be worth using FLP_OPT or FLP_TRACK (depending on your interface - look in the manual) to reduce the maximum track number. A couple of disk utilities go the opposite way, formatting a few extra tracks which most drives can reach.

I returned from Italy with one of Ergon's HD disks, formatted to 1.52 megabytes by their new disk utility. We both have the official Miracle drives and it works fine, but if tracks are going to fail the inner ones are second only to track zero in their vulnerability.

The top limit varies depending on the mechanism of your drive. It has been many years since drives used a spiral groove to move their heads, and spat them out onto the hub if stepped too far! You are unlikely to do any damage by stepping beyond track 79, but the manufacturers would not recommend it.

It's a Lapwarmer, too

Plug in and go

Michael Jonas of NESQLUG has found yet another use for his QL

Working in an industry where "IBM compatibility" is as common a phrase as "Good morning", and a Macintosh Apple is no longer just a snack rolling out of one's raincoat pocket, it gave me great pleasure to discover a feature that my QL has over these modern computing powerhouses. I was sitting on my couch, blanket draped over me to ward off the chill in the air that had seeped into my living room, when I decided to do some work on my QL. After placing the monitor on a small stand beside my couch, and plugging it, as well as the QL power supply, into a nearby wall outlet, I placed the QL on my lap ready to proceed with the task in hand.

Sitting in a cold room makes life difficult for someone who is trying to be creative. I had been sitting on my couch for nearly five minutes, before I even started Quill. A letter had to be written, and the QL seemed more than a match to the task, but the cold was hampering my creative process.

Having used Sinclair com-

puters for many years, I have come to accept a common trait among all - Sinclair computers always generate a lot of heat. This was true with the ZX81. In fact, it got so hot that the computer would usually cease to function after several hours of work. Now personally, I have never had that problem with my QL, although I've heard others complain. The QL does, however, get very warm after about an hour of operation, which always concerns me, especially if I'm in the middle of important work.

Getting Hotter

This overheating has been seen as a disadvantage of the QL. "Bad design" and "poor manufacturing quality" have been used to describe this bothersome feature of the original black machine. But I found out, sitting in my cold living room, that it is actually a hidden feature of Sir Clive's wonderful machine. Now, if you place the QL on your lap as you are doing work, the extra heat that is generated

makes for a nice lap warmer. Why, a hot water bottle can't sustain heat for as long as a QL stays warm. In fact, someone ought to market this idea. I can see it now, an appliance that not only warms your lap, but can be used to write your letters, balance your check book, and play some games when things get too boring.

Imagine trying to do that with an IBM or Apple! Laptops don't generate very much heat, and neither does a standard PC keyboard. You would have to put the entire chassis of a desktop PC on your lap to come close to the heat output of the QL. This would not only keep you shivering, it would probably give you a hernia. Just try to picture someone balancing a full desktop PC on their lap and typing a letter, straining under the excess weight. If this doesn't kill one's creative juices, then nothing will. Why then spend all that money on a personal computer? Just grab a slab of granite and a chisel and have at it.

This idea of computers as appliances is not as ridiculous as it sounds. Most major computer manufacturers, including IBM and Apple, are saying that 10 to 20 years down the road, most computers won't be desktop models that people will sit at to key in information. The computer industry is currently steering in the direction of smaller, more portable machines with multi-processor and multi-tasking architecture that will be as easy to use as today's VCRs - just plug it into your tv and go, as easy as any other household appliance.

Sounds to me like the industry is striving, as usual, for what the QL already has. The QL is compact and portable, and can easily be plugged into your tv for easy computing. And it already has a multi-tasking operating system, as well as a dual processor architecture. Add to this its lapwarming capabilities, and nothing comes close to beating this bundled appliance in price and performance. I guess Clive Sinclair knew what he was doing when he named this machine the Quantum Leap.

QMATHS TWO *The real thing*

Hilary Snaden runs the second part of Digital Precision's maths collection.

This review of the second part of Digital Precision's mathematical software package got off to a less than ideal start when the first line of the boot program crashed my QL. The culprit turned out to be MANDEL_CODE, a file which was loaded as an extension when it was in fact a machine code routine loaded by one of the programs at run-time.

Once the unneeded RESPR-LBYTES-CALL sequence had been removed there were no more problems with the other extensions on the disk - Turbo and Qliberator runtimes, a segment of Toolkit II (not needed if TKII is already available) and QLVAL, two extensions to Superbasic which were also on the first Qmaths disk.

Mandelspeed

The first program on the boot menu is Mandelspeed which, as its name suggests, draws fractal images.

To an earlier generation, "new" mathematics meant Boolean algebra and calculus. Today's new mathematics is concerned with strange attractors, chaotic dynamics and fractals. The concepts may be even more difficult to grasp than those of calculus, but the results are certainly more interesting and relevant to the real world. The opening

of this important scientific avenue is largely due to computers, which are well-suited to the rather mindless, repetitive number-crunching which fractals involve.

A number of Mandelbrot programs for the QL have appeared over the years, and one would expect a commercially-available program to have more to offer. The menu screen suggests that this is the case with Mandelspeed.

Once an initial memory allocation has been made by specifying the maximum size (limited only by available memory) of data table generated, the menu options can be selected either by pressing an initial letter, or highlighting via the cursor keys, and pressing Enter. The program is somewhat sluggish in responding to Enter, but once it has caught on, it will in most cases cycle through the available sub-options, or present a reasonable default.

Inkblots

Besides the normal co-ordinate data (defaulting to the location of the familiar "inkblot" pattern) and iteration count (variable from 8 to 250) there are a number of other options.

The size of the image (or more accurately the size of the table from which the image is generated) can be defined, up to a maximum set when the program first

starts. 100 produces quickly a small "test run" picture, and 250 a full (or nearly full) screen image. Larger sizes generate a larger table but no corresponding screen appears while the data is being generated.

There would seem to be little point in setting the table size larger than 250 since, although a larger table can be displayed, the extra detail will not be readily apparent. It may be possible for a custom printer-driver to use a large data table to generate high quality hardcopy, but this is not explored in the current version of Mandelspeed.

The screen can be switched off to run Mandelspeed as a background task while it is generating its "virtual screen". This is needed partly because the program appears to assume the current screen is at 131072 and POKEs to it; consequently, the screen needs to be disabled even when Mandelspeed is run under the Pointer Interface.

The machine code maths routine MANDEL_CODE reads the keyboard directly (via MT.IPCOM) for an abort instruction rather than the program's CON channel. Perhaps this is to avoid slowing the execution of the maths routine, but it makes background multitasking fiddly, as the keys (usually Esc) used to abort the calculation loops will probably need to be changed to avoid collision with other programs.

Three Modes

The documentation warns of the danger of "losing" the cursor if Ctrl-C is used to switch to another task while on the menu page. Maybe an active cursor on the menu page would have avoided this.

Three different Mandelbrot configurations and the Julia set can be selected, and three calculation modes: Coarse, Two Pass and Slow, successively generating more accurate pictures more slowly.

A point can be "targeted" for use with the Julia and Twisted Mandelbrot displays, either by entering the numbers directly or by moving a crosshair over the current picture.

The Draw and Enlarge options generate a table and picture from the program's current settings, or, if the settings are unchanged, redraw from the current table, which, not surprisingly, is considerably quicker. Enlarge goes on to add a rectangular cursor which can be moved over the current picture, and shrunk/enlarged, to select an area to be recalculated in greater detail.

Both the Enlarge cursor and the target crosshair are manipulated via the cursor keys, which are somewhat tardy. It would be useful to have a co-ordinate display onscreen while the cursor moves.

The Colours option

enables each iteration band to be individually recoloured, and it is easy to restore the "original" colouring. This enterprising facility would be even better if the colours could be paged through, rather than having to scroll through up to 250 in sequence.

ations, size, calculation mode and target co-ordinates. By saving both the table and the data which generated it, they can later be reloaded for further investigation.

The size of the data file depends on the table size; a table of dimension 250

load it back again, but also would not properly allow the entry of new co-ordinates from the keyboard. The program now contains a perfectly accurate representation of a previously-saved table, but it has no idea of how it got there! This makes it practically

impossible to re-use, since Enlarge will generate a different image from the one expected.

This needs seeing-to. If at the full data was saved and loaded with the table, it would make the program easier to use, since the table is not much use without the data.

You can also save and load the colour map which has been defined from the Colour option.

There is a screen-save option for graphics printing programs, though for some reason

a saved screen is a byte short of the full 32768, which can cause problems.

The remaining file options set the data device and filename and obtain a directory listing.

In fractals there is something for everyone. For the programmer, the challenge that however quickly and accurately the calculations are made there is always

more detail tantalisingly out of reach. For psychologists, philosophers and artists there is the striking analogy that the most fertile areas of the image are at the very borders of the unconscious central dark and the conscious outside areas. For those in pursuit of life's aesthetic pleasures, there is the inexhaustible beauty of the images themselves.

This beauty is apparent even with the QL's relatively modest display in Mode8, and Mandelspeed could have a great deal to offer the fractal enthusiast if the snags found in my review copy were expunged.

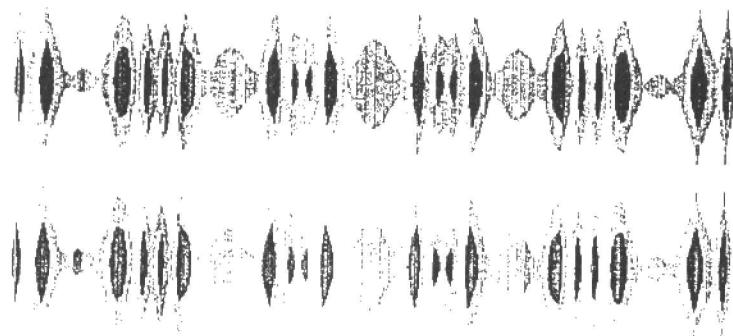
3D Terrain

3D Terrain draws three-dimensional landscapes. Exactly how the basic pictures are produced is not revealed by the documentation, and the user cannot add to the twenty already supplied.

A large number of options on the main menu allow parts of the "landscape" models to be selected and manipulated. The angle and distance of the viewpoint, and so the perspective, can be altered, the vertical axis expanded, and the resolution adjusted to produce a rough-and-ready outline or a detailed picture. Screens can be drawn in Mode4 or Mode8, and saved to a file or dumped to a printer.

As with Mandelspeed, the most interesting pictures are a long time a-coming, but unlike Mandelspeed you cannot write a "virtual screen" as a background task while the machine does something else.

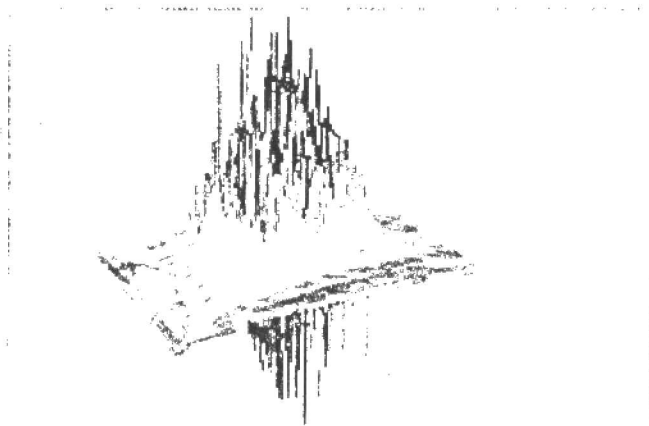
3D Terrain's development on a 128K QL shows considerable ingenuity on the part of the programmer, but unexpanded QLs are rarer than they once were, and the program is certainly unusual and interesting enough to be worth developing a version which can



$f(x,y) = \sin(2x) + \cos(y)$ $R=[-4, 4]$ $p1=[-2, 2]$ $p2=[-2, 2]$ $z=[-1.97776599, 1.99675065]$ 80×40
Nombre del archivo donde guardar -> Flp2_example_scr

Above: GLNIVEL SCREEN. (inverted colours)

Below: 3D TERRAIN SCREEN. (inverted colours)



SCR:50010BJ:1001Y:-301P:15
"Big Bang"

Save and Load

A number of save and load options are available. "Table" saves or loads the complete "virtual screen" table, together with a file containing its size and the number of iterations. "Data" saves or loads the image co-ordinates, number of iter-

will produce a data file 62500 (250x250) bytes long. The save options check that there is sufficient room before writing the file.

The savable and loadable table is an excellent idea, unfortunately hampered in the review copy because it did not work. Co-ordinate information seemed to be saved accurately, but the program not only refused to

use available memory to hurry things along, or at least use Qdos's multitasking through the "virtual screen" approach.

The program would be more flexible as a single executable task; in its present form it behaves oddly in the presence of the Pointer Interface, which is gaining popularity.

A short Basic program supplied illustrates some of the principles involved in 3D Terrain, but the best way is to experiment. A certain amount of patience is needed.

QLVAL et Al

The QLVAL SuperBasic extensions were first introduced in Qmaths 1, where they were used in the Surface Modeller program. Here they are used in three further programs, Qalc, Numint and Glnivel, and properly documented in their own right.

When loaded with either a RESPR-LBYTES-CALL sequence or an LRESPR command, QLVAL adds two new functions to SuperBasic, CMPILAS and VAL. These provide SuperBasic with a facility analogous to VAL in other Basics, so that mathematical expressions can be evaluated in the form of text strings.

QLVAL does this in two stages to maximise speed. First, the expression to be evaluated is fed to CMPILAS, which returns an equivalent string to be stored as a Superbasic variable. This can then be evaluated for specific values of up to five variables in the original expression (designated x, y, z, u, and v) by feeding it to VAL along with the appropriate values of x,v.

In this way the relatively slow process of converting the human-readable text expression into a form which the QL can more

easily understand only needs to be done once for each expression.

The two functions are used something like this:

```
f$="asin(sin(x)sin(y)+cos(x)
cos(y)cos(z))"
t$=CMPILAS(f$)
y=RAD(52):z=RAD(37)
P R I N T
VAL(t$,0,y,z)
P R I N T
VAL(t$,PI/2,y,z)
```

Unset variables (in this case u and v, not actually used here) default to 1.

The CMPILAS function is more flexible in its syntax checking than SuperBasic, so expressions can be entered in a style which mathematicians would find more familiar, and the extensions are likely to be useful to anyone writing SuperBasic programs which need the input of mathematical functions.

For assembly language hackers, the source files, with copious commentaries (in Spanish!) are supplied archived on the Qmaths disk. Users tinkering with the code are requested to inform Digital Precision.

Qalc

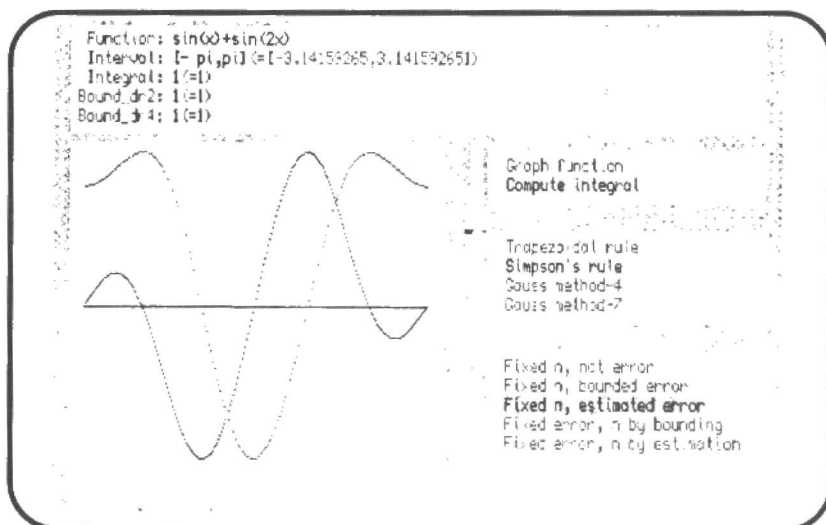
Qalc is a compact scientific calculator, accepting input as expressions in the text format expected by CMPILAS function. The results of the three most recent evaluations are stored in variables x, y and z, which can be recalled and used in other expres-

sions. Two further variables can be user-defined.

Some expressions, for example $\sin(\pi/2)$, can produce confusing results. This is due to rounding errors introduced by Qdos' maths routines rather than the coding of this neat and efficient program.

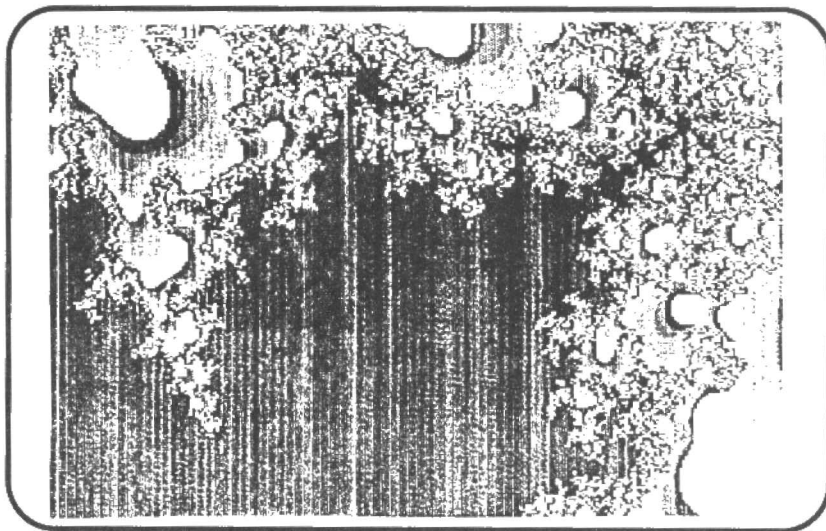
(if known) the exact value of the integral and the upper bounds of the second and fourth derivatives.

You can then select from the Trapezoidal rule, Simpson's rule, or two variants of Gauss' method, whether or not to plot the integral as well as the origi-



Above: NUMINT SCREEN. (inverted colours)

Below: MANDELSPEED SCREEN.



The restore-screen facility and the presence of an active cursor means that Qalc can be multitasked with the minimum of bother.

Numint

Numint calculates and plots the integral of a single-variable function over an interval. The function, bounds and number of points are entered first, then

nal function. Various permutations of fixed, bounded and estimated errors and values of n are offered.

F4 toggles a trace mode which displays intermediate values of the function for fewer than fifteen points, and F5 brings up a miniature scientific calculator.

The captions "With integral curve" and "Without integral curve" seem to have become transposed, but apart from that the oper-

ation of the program is exemplary.

Numint looks neat and professional, is so straightforward it is almost self-explanatory (using almost exclusively up, down, Enter and Esc.) and very fast.

Glnivel

Glnivel plots the function $f(x,y) > z$ over a rectangle in the plane OXY for a real value of z . The options available are not at all apparent from the program, and even with the help of the documentation I needed a fair amount of experiment to make them work properly.

Once the function, the corners of the rectangle and the required resolution in x and y have been entered, press F4 to enter a value of z for which x and y will be plotted as a line or points, or Shift-F4 (not Ctrl-F4 as stated!) to enter a value of z and a colour to fill a region $f(x,y) > z$. F1 restarts the program, Shift-F1 prompts for a new function, F2 a new rectangle and resolution and F3 a new resolution.

Since there is no error checking of data input it is extremely easy to bomb out of Glnivel with a Turbo error message, and, like the Surface Modeller program on the first Qmaths disk, very little coding has been spent on presentation. Many users might feel intimidated by a program which starts with a screen wholly blank apart from the prompt " $f(x,y) =$ " in the bottom left corner, and even adept mathematicians can make mistakes during data entry.

Both this program and Surface Modeller produce good quality graphics, but both cry out for a more friendly presentation. Numint and Qalc demonstrate that mathematical programs can be functional and look good.

The Superbasic sources

of Qalc, Numint and Glnivel are in an archived file.

Acalc and Rpcalc

Acalc contains the maths routines for the high precision calculators in Qmaths 1. An extension which makes the routines available to SuperBasic via the keyword ACALC, and the assembler source file, are supplied.

Rpcalc is a configurable version of the high precision calculator. The Basic source, and a compiled version and SuperBasic configurator, are supplied.

A good working knowledge of SuperBasic, Qdos, and assembly language programming is needed to make the most of these files.

Abacus Compendium

By far the largest file on the Qmaths 2 disk is an archive containing a number of files which would otherwise not fit on a 720K disk. These can be extracted from the Extract option on the boot menu, by running the program UNPACK_BAS, or by using directly ARC as supplied on the disk.

Among these files are over 500K of Abacus spreadsheets. There is little documentation on the 26 _aba files, and from standard deviations and confidence intervals to multiple linear regression and beyond, they are for advanced statistical analysis. Those to whom Kruskal-Wallis, Mantel-Haenszel, Mann-Whitney or Kaplan-Meier mean anything may think they have struck gold.

Overview

It is difficult to avoid commenting on the packaging of the two Qmaths disks.

Here are two enterprising and ambitious fractal programs which in some ways represent the cutting edge of modern maths while offering much on aesthetics too; a number of programs dealing with more traditional calculus: and one program which seems to fall between the two. Here is assembler code for mathematical manipulation and some tools for highly advanced statistical analysis.

Those whose interests lie mainly in exploring fractals, or in traditional calculus, or in statistics, might feel aggrieved at having to buy both Qmaths to get what they want.

Users of maths software are likely to want to multitask the most useful programs. Unfortunately this is difficult because of the varying degrees to which the programs tolerate the Pointer Interface (which is the most straightforward way of multitasking conveniently).

All the programs are perfectly usable, but I cannot help feeling that the valuable ideas in Qmaths could have been made even more practical by attention to this point.

Matters Arising

It is possible to view the final, "official," Sinclair releases of Qdos as the last word in QL firmware, warts and all. On the other hand there is an argument that if the QL is to survive and grow, Qdos must find a way of continuing to develop in something like the way which its original designers planned.

Minerva is probably the best-known example of this thinking turned into action, and **Qptr** and **Qpac** can be viewed as optional but no less integrated developments of the original operating system. We are now

within view of **SMSQ** running on the QXL as well.

It is not in the least compulsory that software should make use of such "extras" as **Minerva's** second screen, or the **Pointer Interface**, but it is disappointing when new releases clash with them.

The QL community has every reason to be grateful to Digital Precision for their support, and for opening up the kind of deeds that the QL can be persuaded to do, but time and tide are rushing on, making new demands on programmers.

Anyone wishing to find out more about fractals and related ideas may like to investigate the following books:

Ian Stewart: **Does God Play Dice?**

James Gleick: **Chaos.**

Hans Lauwerier: **Fractals.**

Benoit Mandelbrot: **The Fractal Geometry of Nature.**

(Editor's note: We can't guarantee that these titles are all currently in print, but they should all be by ordering from a public library, even if all the details are not known.)

QMATHS TWO

Very Basic SuperBasic

Part 6

Dilwyn Jones uses some colourful language this month!

The QL is blessed with a pretty good range of easy graphics commands. This month, I'll introduce simple colour control and general screen handling. These facilities have been enhanced on new hardware - systems such as the QXL and Atari ST emulator have added new screen resolutions and so on - but for now we shall ignore these in favour of the facilities built into the humblest unexpanded QL.

Screen Modes

The QL has two screen modes. These can be seen on first starting up the QL, though they may not be obvious! One mode gives us a choice of up to 8 colours, while the other gives only 4, but with a greater amount of detail possible on the screen display.

What could be simpler than the command to choose between them. Mode4 switches the QL into the four-colour mode, while Mode8 switches to eight-colour mode. As new hard-

ware gives better graphics facilities, it seems likely that there will be other values which may be used. Minerva system users already have extended mode commands to choose a variety of options.

Mode4 gives us only black, red, green and white to choose from, but Mode8 also adds blue, magenta, cyan (a light blue colour) and yellow. All of these colours have numbers which denote the colour value.

- 0 - Black
- 1 - Blue
- 2 - Red
- 3 - Magenta
- 4 - Green
- 5 - Cyan
- 6 - Yellow
- 7 - White

In Mode4, if you ask for a colour which is not available, the nearest value is used instead, so blue becomes black, yellow becomes white and so on.

There are two types of colour, background colour

and foreground colour (used for text). The background colour is selected by using the PAPER command, while the foreground colour is specified using the INK command, two rather nicely descriptive command names. So if we wished to have text in white ink on red paper background, we would use the commands PAPER 2 and INK 7.

Specified like that, they refer to the area of the screen which is red when you start using the QL. The screen is divided into three areas, or boxes, at first. These are referred to as 'windows', a rather apt name for part of a glass screen!

Channel Numbers

Each 'window' has a number which refers to it. In common with most graphics facilities, numbers are important here. The number which refers to each window is called a 'channel number' and is normally written into colour and graphics commands by preceding it with a 'hash' character #. This is

the Shift character above the number 3 on the English QL keyboard.

This symbol tells us that the number which follows is a 'channel number', not a colour number, for example. The # symbol is important, since it can be omitted in many cases and the computer can only tell the difference between a channel number and a colour number when the # symbol is present.

The three areas are referred to as channels 0, 1 and 2. Channel 0 is the black part at the bottom of the screen. Channel 1 is the red part of the screen, and channel 2 can be blue when the QL is used in TV mode (Mode8) or white in monitor mode (Mode4). If the channel number is omitted in a command, the QL normally assumes you are going to use channel #1. This is referred to as the 'default' channel.

PAPER 1 means use blue paper in channel number 1, while PAPER #2,1 means blue paper in channel 2. INK #1,7 means white ink in channel 1, while INK 7 also means white ink in channel 7.

Channel CLS

These commands start to become really useful when used with PRINT and CLS commands. When you use a CLS command to clear the screen, you can also use channel numbers to refer to

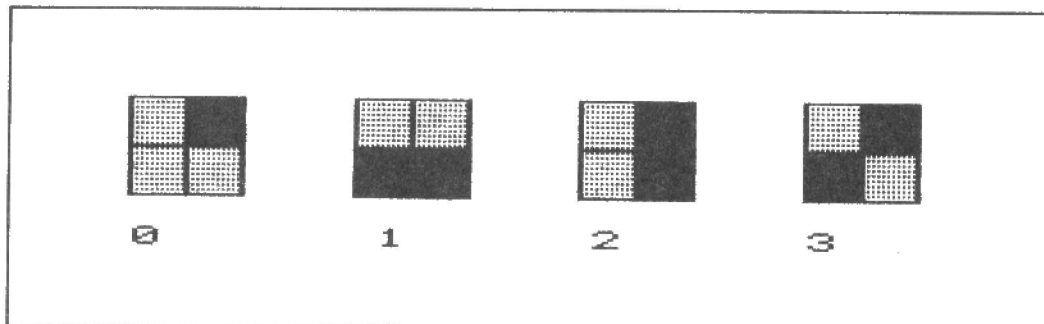


Figure One: Stipple pattern numbers.

which part of the screen you wish to clear. CLS clears that part of the screen to the last colour specified in a PAPER statement.

Therefore, PAPER #2,0 : CLS #2 makes the listing channel (window #2) clear to the colour black. We can

So it can be used to highlight certain print statements, for example. The syntax is the same as the PAPER command. STRIP #0,4 makes the printing to channel #0 (normally the bottom of the screen) appear on a green background without

Colour Spec

You can give the INK and PAPER and STRIP commands two types of colour numbers. You can specify the value as a number from 0 to 255, eg INK 255. You can also specify the colour value as three numbers indicating a colour, a contrast colour (not easy to explain without recourse to binary arithmetic) and a pattern number. The pattern number is a number from 0 to 3. Given a 2x2 block of colour, the values represent dots used as follows. Pattern 0 means use one of the four dots, pattern 3 means use opposite corners in the same colour, while 1 means horizontal strips of colour and 2 means vertical strips. These are best represented by a diagram, so see Figure one. The use of fine dots to create the illusion of extra colours is called Stippling. Colours created in this way are called Stipple Colours.

We can express a black and white dotted colour as either a value of 255, or as the three numbers 7,0,3. At this stage, you have to find the colours by experiment or by looking at lists since there is no easy way to describe how they are made up and what each colour will look like. Figure two shows a simple program which runs through all colour values on the screen, showing the colours and what they look like. It asks you for a mode number, then lists all the colours as strips of colour, scrolling the slowly up the screen - the PAUSE 50 statement waits for one second between each one to give you a chance to look at them while they slowly appear. If you press a key on the keyboard, the program will run a little faster if you wish. Or you can vary the value after PAUSE to change the speed reduce it to make it go faster, while increasing the value will make the program run slower. The value after

pause works in units of 1/50th second.

BORDER

It is also possible to change the colour of the frame around each window, or to have no frame at all. The BORDER command allows you to set the colour and size of the frame around each window, by specifying a channel number (default value if omitted is #1 as before), size of the border and the colour value, which can be a single number or three numbers as discussed above. BORDER #2,1,4 puts a green border which is 1 dot high at the top and bottom (it is actually two dots wide at the edges for reasons we won't go into here) around window #2. You have to be careful to specify a size which will actually fit into the window, or the QL will give an error report!

Figure three is a colourful example showing how to use the BORDER command to generate some pretty patterns by changing the size of the border. It looks best running in Mode8, but will also run in Mode4. It flashes a lot when running; be careful if you are prone to epilepsy through visual disturbances.

It would be nice if we could place blocks of colour inside a window as well as writing text in colour. There is a command called BLOCK (surprise surprise!) to do this, but to use this we need to learn about pixels first.

Pixels

The 'screen picture is made up of hundreds of tiny dots or picture elements, called pixels. Each is just large enough to see on a good display monitor. Letters and graphics drawn on the screen are made up of combinations of these little dots. In mode 4, there are 12 dots

```
100 PAPER 0 : INK 7 : CLS
110 INPUT 'Mode number (4/8)?';mode_no
120 MODE mode_no
130 CLS
140 FOR colour = 0 TO 255
150   STRIP colour
160   REMark 10 spaces in next line
170   PRINT '          '
180   STRIP 0 : REMark black
190   PRINT ' Colour ' ;colour
200   PAUSE 50
210 END FOR colour
```

Figure two: Print colour number examples.

```
100 colour = 0
110 FOR size = 50 TO 0 STEP -1
120   BORDER size,colour
130   colour = colour + 1
140 END FOR size
```

Figure three: Using BORDER to make patterns

```
10 PAPER 0 : CLS
20 FOR y = 0 TO 100 STEP 20
25   FOR x = 0 TO 100 STEP 20
30     BLOCK 20,20,x,y,RND(1 TO 7)
35     PAUSE 5
40   END FOR x
50 END FOR y
```

Figure four: BLOCK demonstration.

use PAPER to set the background colour for a print statement like this, although there is also another command to help us with this.

```
PAPER #2,0
CLS #2
PAPER #2,2 : INK #2,7
PRINT "White on red"
```

The STRIP command is like the PAPER command, but has no effect on the paper value used by CLS.

affecting the colour of the whole window. When you use the PAPER command, it automatically makes the STRIP colour the same as the new PAPER colour until you change it.

Although the QL has only eight basic colours, it is possible to combine these colours in fine dots or lines to give the effect of further colour patterns, up to 256 in total.

across the screen and 256 downwards. In Mode8, there are only 256 dots across, which is why everything looks a bit more chunky in Mode8. Because there is twice the number of colours, half the number of dots are available in order to ensure that the computer uses the same amount of memory to keep the picture.

The screen is always addressed as the number of dots across and down the screen, 0 to 511 across in both modes (to keep things equal to the programmer) and 0 to 255 down. The origin is position 0,0 at the top left hand corner of the screen.

Few QL commands use the absolute value across the screen. The BLOCK command uses the co-ordinates from the top left corner of the window it refers to, not from the top left corner of the screen. To place a block of colour on the screen we must specify the width and height, along with where in the window to place the block of colour, and the colour itself, which can be different to the ink and paper colours.

BLOCK #channel
!width,height,x,y,colour

The channel number can be omitted if required, in which case it refers to the default of channel #1. "width" is the size of the block (the maximum is 512 in both modes, though it depends on the size of the window), "height" is the number of dots high (up to 256, although it depends on the size of the window). "x" and "y" are the distance from the top left corner of the window. Figure four helps to show how this works - it draws a block of random colour which is placed at progressively greater distances from the top left corner, first of all moving to the right, then going back to the left a little further down until it reaches the final position.

Experiment a little with all of the values to see what effects you can get and see

DOW command to specify where an area of the screen is to be placed and what its

the dimensions used to see how they affect the size and location of the window.

```
100 REMark alter #0, #1 and #2
110 MODE 8
120 REMark first clear the entire screen to black
130 WINDOW 512,256,0,0
140 PAPER 0 : CLS
150 WINDOW #0,448,42,32,16
160 PAPER #0,4 : INK #0,0 : CLS #0
170 BORDER #0,1,2
180 PRINT #0,'This is window #0'
190 WINDOW 448,82,32,70
200 PAPER 2 : INK 7 : CLS
210 BORDER 1,255
220 PRINT 'This is window #1'
230 WINDOW #2,448,82,32,160
240 PAPER #2,6 : INK #2,1 : CLS #2
250 BORDER #2,1,1
260 PRINT #2,'This is window #2'
```

Figure five: WINDOW demonstration.

if you can work out how to draw a box around a piece of text, for example. (Hint: use the AT command to position some text and see if you can work out how many pixels it takes to be the same size as a piece of text and how you can make BLOCK locations match AT locations).

It is very important to stress that although there are 256 pixels across the screen in Mode8, the system requires you to pretend that there are 512 like Mode4, but odd values are rounded down to even values, or odd values are ignored. In other words, in mode 8, 1 is the same as 0, 3 is the same as 2 and so on. Values across should be multiples of 2 for both width and the location across.

Moving Windows

Having mastered the principle of co-ordinates and pixels, we can move on to windows themselves. It is possible to rearrange the areas of the screen - we can change their sizes, we can swap them about, we can move them to different places. We use the WIN-

```
1000 DEFine PROCedure TV_WINDOWS
1010 WINDOW #0,448,40,32,216
1020 CLS #0
1030 WINDOW #1,448,200,32,16
1040 CLS #1
1050 WINDOW #2,448,200,32,16
1060 CLS #2
1070 END DEFine TV_WINDOWS
1080 DEFine PROCedure MONITOR_WINDOWS
1090 WINDOW #0,512,50,0,206
1100 CLS #0
1110 WINDOW #1,256,202,256,0
1120 BORDER #1,1,255 : CLS #1
1130 WINDOW #2,256,202,0,0
1140 BORDER #2,1,255 : CLS #2
1150 END DEFine MONITOR_WINDOWS
```

Figure six: How to restore original windows.

size will be.

WINDOW width,height,x,y

Where "width" is the number of pixels across, "height" is the number of pixels down, "x" is the location across the screen and "y" is the location down the screen. Again, "width" and "x" must be even values.

Figure five shows how to rearrange the windows you get when the QL starts up into three horizontal sections one below the other. Vary

Remember that the origin value ("x") plus the width must not exceed 512 and the "y" origin value down plus the height must not exceed 256.

Note how the channel numbers (preceded with #) can be used in PRINT statements and CLS statements to make them operate on selected parts of the screen.

Restore All!

Having reorganised our screen, we need to know

how to put it back together again! Figure six shows the statements needed to reposition the windows for television and monitor mode. These are two routines you can add to your own programs. They are written as procedures - use the name TV_WINDOWS to restore television display sized windows and MONITOR_WINDOWS to restore the monitor display sized windows to cover most of the screen. These two routines are very useful for undoing whatever your program has done to the normal windows! If required, add your own PAPER and INK commands to restore the colours as well as the size of the windows.

You may have noticed that the commands to set up television mode windows use a smaller section of the screen than the monitor version. This is because some television sets cannot display the full 512 pixels

across the QL display, they tend to clip a little bit at the left and right of the picture, so the television mode windows deliberately get positioned 32 pixels from the edge to make sure you can see them all.

There are a couple of other useful commands which should be mentioned here too. If you are using the computer in Mode8, it is possible to make printed text flash on and off, by using the FLASH command. FLASH 1 makes text flash, while FLASH 0 means that the next text written to the screen will not flash. Both can have channel numbers FLASH #0,1 turns on flashing in the bottom window. It does not work in Mode4.

If you would like to be able to print text over a picture or without disturbing some areas of colour you have set up with BLOCK, there is a command to make printing ignore the paper colour.

Unsurprisingly, the command is called OVER and can have three values or states. OVER 0 is the normal setting, which means that text will be printed in the selected ink colour with the selected background paper or strip colour. OVER 1 means 'transparent', or no strip colour is used. OVER -1 gives a rather ghostly effect where printing is sometimes transparent or the colour does not come out as expected. Some binary arithmetic is again required to fully understand what goes on, but it does have one very useful function - if you print something again on top of itself, it makes the original disappear as long as colours did not change. This can be very useful when programming moving graphics on the screen, for example, although you will not encounter it much at first. Like other commands, OVER can take a window

channel number, defaulting to #1 if not specified.

In the next issue, we will be developing our knowledge of QL graphics and printing.

**Very Basic
SuperBasic**

ARCHIVE YOUR QL COLLECTION

Now you can keep your Sinclair QL World magazines safe and clean. No more dog-eared covers or missing copies . . . You can protect your magazines in this high quality, specially-created binder. This Sinclair QL World binder will comfortably hold a complete year's issues of your favourite Sinclair magazine. It is a high

quality product, British-made and comes with full binding instructions. It is manufactured in a rich, deep blue with genuine gold blocked lettering. Enhance your Sinclair QL World Magazine collection now for only £6.95 (inc.P&P!) *Send for one today!* The QL World binders also make an ideal gift for other Sinclair users too!

TO: QL WORLD, THE BLUE BARN, WOOTTON, WOODSTOCK, OXON. OX7 1HA

Please send me ☐ QL World binders - I enclose £6.95 for each binder including VAT. postage & packing.

Readers outside the UK and Eire please add £1.95 for surface overseas mail.

Please make cheques payable to ARCWID LTD.

ACCESS ☐ VISA ☐ account:

Expiry date:

Name

Address

Postcode

Tel No.

QL-PC FILE-SERVER

A low-cost method of hooking PC hardware to the QL, described by user Jack Brown.

INFORMATION

Product: QL-PC Fileserver

Price: £24.50

Suppliers: Written by Di-Ren.
Available from Dilwyn Jones
Computing, 41 Bro Emrys,
Bangor, Gwynedd, LL57 3YT. Tel
0248 354023.

Serial link leads: From TF
Services, 12 Bouverie Place,
London W2 1RB. Tel 071 724
9053.

Di-Ren, the manufacturers of this program, are not prolific, but when they do produce something for the QL I usually find it interesting.

I cite, for instance, Fleet Tactical Command and QL Network Prover.

I guess that most of us who have access to PCs have tried, or at least considered, the possibility of linking the PC and the QL. In the past there have been a few articles on how this might be achieved but invariably these have only been suitable for occasional operations. At last someone has come up with the solution.

PC hardware

The QL-PC Fileserver is a program designed to give the QL direct access to a PC's hardware devices: disk drives,

screen display and printer ports. When installed, the program essentially converts the PC into a fileserver for the QL. From the QL user's point of view, once linked, the PC just looks like any other QL device. For instance, entering DIR PCD3_ will display the PC's drive C: directory in the normal QL format.

The program is full of useful features. Not only can you directly access any device DOS sees as a disk drive, be it floppy, ram or hard disks; you can also access the PC's screen display and, exceptionally, the PC's printer ports. This last feature is very beneficial as it means I no longer have to swap printer leads or get myself a switch box to allow both machines to use the same printer.

Another important feature of

the fileserver is its ability to access PC networks through the linked machine. It would appear that any device that DOS "sees" as a disk drive is also accessible by the QL. To test this, I dragged the QL into work and tested it over our Novell network. It lived up to the authors' claims, of course!

Links

Connection of the two machines is via the serial links. Di-Ren supply a serial pin-out table but, unless you have some knowledge of serial linking, it would probably be advisable to obtain a ready-made lead. These are available from TF Services. Having a suitable lead on

QL CONNECTIONS

PC CONNECTIONS

	QL UK BT 6 PIN TYPE	QL 9 PIN D TYPE	PC 9 PIN D TYPE	PC 25 PIN D TYPE	PC 9 PIN MCOM	PC 25 PIN FCOM
GROUND	PIN 1	PIN 1	PIN 5	PIN 7	PIN 5	PIN 7
TXD	PIN 2	PIN 2	PIN 3	PIN 2	PIN 3	PIN 2
RXD	PIN 3	PIN 3	PIN 2	PIN 3	PIN 2	PIN 3
DTR	PIN 4	PIN 4	PIN 7	PIN 4	PIN 7	PIN 4
CTS	PIN 5	PIN 5	PIN 8	PIN 5	PIN 8	PIN 5

For QL Serial Port 2, swap QL pin connections 2 and 3, swap pins 4 and 5.

DISCLAIMER

No claim is made as to the suitability or accuracy of these instructions for any purpose. We do not accept liability for any damage to machinery or data that may occur in connection with the instructions contained herewith.

hand from my previous experimentation it was just a case of plugging it in and trying out the program.

The PC part of the program arrived on both 5.25-in and 3.5-in formats and is in the form of a TSR (Terminate and Stay Resident program). The ability of the TSR to operate as a background task is of particular significance. Not only does it look after the QL's interests, but also allows you to run PC software at the same time.

Invoking the PC part of the program requires a short command line in the context of QLNET /Comm port/Baud Rate/Default Drive/V2. Comm port is the Comm port number you are using to link the machines. Baud rate should be the highest baud rate common to both machines. Default drive is either a ram drive or your default disk drive (normally C:). V2 is optionally invoked if you require the program to be a stand alone task.

For instance, QLNET /COM1/9600/C: will invoke the TSR using Com1 to link the machines at a baud rate of 9600. Drive C: is the programmes working drive.

The QL program

The QL program arrived on a 3.5-in DSDD disk. Contact DJC if an MDV version is required.

The program is loaded in the same way as any toolkit: LRE-SPR FLP1 _NET_BIN (if you have TKII), or locate=RESPR (6200). LBYTES FLP1 _NET_BIN/locate. CALL locate. This loads the toolkit. To invoke the fileserver first set the Baud rate and then enter PCSERVE Ser(?), where ? is the serial port number you will be using. A point to consider before starting the program is that TKII's command PRT_USE, must not be invoked to the serial port you are using to link the machines. This sets up output buffering to the port and the file server doesn't like it! Make sure that this command is not

embodied in your boot files.

Unlike many QL software manuals in my possession the fileserver instructions are refreshingly clear and concise. An explanation of each command is given, as well as an example of correct usage. It's a pity more instruction manuals aren't written like this. There is nothing more infuriating than having to wade through reams of useless waffle only to find there is no specific example of the command in question.

File handling

Accessing files stored on PC disks is done in exactly the same way as native QL devices. Typing the command PCLIST returns the drives available. Two directory device drivers are available; PCD and PCX. Both are used in the normal QL style. For example, OPEN NEW#10,PCD1 _TEST_FIL will open a new file on the PC's floppy drive A. PCD is the driver you would normally use. PCX automatically converts document newline codes for files you may wish to edit on either machine. Again, I have tested file operations with the well-used QL programs Quill, Abacus etc., and there are no problems. Another major plus for the program is its ability to recognise native QL files stores on the PC. This means that it is possible to EXEC QL programs direct from the PC.

Subdirectories on the PC can be made and removed using the commands PCMD and PCRD. Subdirectory stepping is achieved by using TKII-style commands PCUP, PCDOWN and so on. It is also possible to manipulate the PC file read/write attributes directly from the QL.

Most standard and TKII wild-card file-handling commands are accepted by the fileserver. You can for instance DIR, WDEL, WSTAT and so on. Not supported are FORMAT, RENAME and TRUNCATE.

Transfer speed

The speed of data transfer is the program's only drawback. It is necessarily limited by the QL's slow serial port handling capabilities and cannot be compared to loading from floppy or hard disks. However, since most Qdos files are quite compact (especially compared to PC files) this isn't a problem.

After several tests I calculated an average transfer rate of around 750 characters per second. This is transferring data from the QL's ramdrive to the PC's (486) hard disk, with the baud rate set at 9600. If you have the 8049 replacement, Hermes, fitted, data transfer speed can be dramatically increased due to the ability of Hermes to transmit and receive at 19200bps.

A point worth noting is that it appears to be far quicker transferring from the QL's ramdrives than from floppies or even worse, microcassettes.

I now make a practice of copying files from the PC to the QL's ramdrive before manipulation. This is not only far more efficient, it also means there is a back-up copy should I make a mistake.

The Drivers

When the fileserver is invoked the QL has a new device driver "PSCR". This opens a text window on the PC's display (similar to the QL SCR device). It is a simple driver that supports PRINT, AT and CLS commands and is useful if you wish to view two pages of information at the same time.

A much more useful feature is the device driver LPT. This works in exactly the same way as sending printer data to the SER ports, except that data is actually presented to the PC's LPT ports. I have extensively tested this and it works perfectly. Output to the QL's SER ports (including the port used to connect the two machines!) can be

re-directed using the command PC_DEV, which means you don't have to re-configure your programs.

The combined PC and QL program package costs £24.50 - not bad value for money. It is not only novel to be able to connect almost any QL and PC together in a useful manner, it is also a very economical way of directly accessing the PC's hardware, in particular its very large mass storage capabilities.

The program will run on all QLs, with or without toolkits, and PC-compatible XTs and ATs that are running a minimum DOS version of 2.2 (practically any modern PC, in other words).

I have found the fileserver to be very useful. All my important QL files are now archived on the PC and the printer is permanently accessed through the fileserver. No more fiddling!

Serial Links

Making up Serial link leads for UK QLs with BT-type 6 pin plugs is something of a tedious business without the assistance of a special termination tool. If you do make up your own leads, it is essential that you check out the finished product with a Multimeter or some other type of circuit tester, and that the connections are correct, before using them.

The following information is from Di-Ren's Serial Link Pin-Out Table (Issue 3), which you can get with Fileserver in any case:

25- and 9-pin connectors are normally numbered on the face of the plug. The UK-type BT plug used for the QL has Pin 6 nearest the latch.

Pin 6 on the QL BT-type plugs, and pin 9 on QLs with 9-pin plugs, carry a 12-volt DC current from the QL. Under no circumstances should these pins be connected. Di Ren's connection table is reproduced as Figure one, but please note that connections vary slightly in different countries, and you should do your best to check that the connections are correct.

DIY TOOLKIT

Simon Goodwin translates his DIY Mouse driver into interrupt-driven QL machine code.

In last month's QL World I explained the messages sent by two types of widely-available serial mouse, and decoded the signals via Hermes, SuperBasic and the QL serial port. This version includes a machine-code interrupt server which runs fifty times every second, translating incoming data into co-ordinates and button information which can be manipulated from SuperBasic.

Loading

Listing One is the assembly code, developed and tested with HiSoft's Devpac 2 assembler. Type this into your own assembler if you want to modify or experiment with the source. The only unusual features are the word sizes after vector addresses like \$E2.W (IO.QOUT) and \$11A.W (BV.CHRX). Remove the second .W if your assembler objects to these lines, and all will be well. The workings of the assembly code are discussed, as usual, at the end of this article.

Listing Two generates code for the two-button SER2 Microsoft serial mouse handler from Basic DATA statements at the end of

```
* QL WORLD DIY TOOLKIT - MOUSE/POINTER DRIVER
* Links SuperBASIC to a 2 or 3 button PC serial mouse
* Version 1.6, Copyright 1993 Simon N Goodwin
*
ser_port      equ      2           Configure to one or two
ser_pointer   equ      148+ser_port*4 152 for SER1, 156 for SER2
buttons       equ      2           Configure to two or three
*
* Configuration error trapping for DEVPAC etc.
*
        ifne      buttons-2
        ifne      buttons-3
        fail
        endc
        Wrong number of buttons
        endc
*
        ifne      ser_port-1
        ifne      ser_port-2
        fail
        endc
        Serial port 1 or 2 expected
        endc
*
* Offsets of variables in interrupt linkage
*
link          equ      0           Interrupt link comes first
vector        equ      4           Offset of code address
marker        equ      8           Offset of "DIY1" signature
prefix        equ      12
*
* These variables are relative to PREFIX
*
latest_x      equ      0           Current co-ordinates
latest_y      equ      2
limit_x       equ      4           Right margin limit
limit_y       equ      6           Top margin limit
step_x        equ      8           Counts per horizontal move
step_y        equ      10          Counts per vertical move
button_bits   equ      12          Bits shadow each button
synchro       equ      14          Input byte # 1..3 or 1..5
initial       equ      15          Microsoft initial byte
serial_id     equ      16          Zero or serial channel ID
var_end       equ      20
var_length    equ      prefix+var_end
*
start         lea      define,a1
              move.w   $110.w,a2    BP.INIT vector
              jmp      (a2)
*
* PTR_ON asks is it already on? if so, return no error
*
point_on      bsr.s    find_pos2
              beq.s    it_worked
*
* Otherwise, allocate linkage area in common heap
*
              moveq    #var_length,d1
              moveq    #0,d2        Owned by SuperBASIC
              moveq    #24,d0       MT.ALCHP trap key
              trap     #1
              tst.l    d0           Did we get it?
              bne.s    oops
              lea.l    server,a2
```

the listing. Type this in and run it to generate the MOUSE_CODE file.

```
X=RESPR(650)
LBYTES FLP1_
MOUSE_CODE,X
CALL X
```

This sequence of commands adds the extensions PTR_ON, PTR_OFF, PTR_POS, PTR_MAX, PTR_INC, PTR_LIMITS, SYNCH%, BUTTON%, X_PTR% and Y_PTR% to SuperBasic. If you don't like the identifiers you can patch their characters with utilities like Spy, The Editor or FEDIT, or re-assemble Listing One with completely new names at the end. For top compatability, load the standard names and use ALIAS from DIY Toolkit Volume A to set up your own alternatives.

The DIY Toolkit mouse driver is included in the latest volume on disk, available from Dr. Bill Fuggle. Volume I (I'm running out of alphabet) includes last month's SuperBasic prototypes, wiring details, example programs, assembler source and binary code for "Mouse Systems PC" and "Microsoft" serial mouse drivers to suit either serial port.

DIY Toolkit volumes cost three pounds each on disk or microdrive cartridge, and come with printed documentation if you order two or more. Twenty five volumes are available from DIY Toolkit, 86 Lordwood Road, Harborne, Birmingham B17 9BY. Send a stamped self-addressed envelope if you would like further details.

```

                                move.l    a2,vector(a0)    Provide code address
                                move.l    d7,marker(a0)    Add signature
                                move.l    #$1FF00FF,prefix+limit_x(a0) * 511, 255
                                move.l    #$6000A,prefix+step_x(a0)   * 6, 10
                                move.l    a0,d6
*
* See if serial channel is open
*
                                moveq     #0,d0              MT.INF
                                trap       #1
                                tst.l     ser_pointer(a0)    Is SER input queue open?
                                bne.s     link_it             Nearly finished
*
* If not, take note & open it; assume correct baud rate
*
                                lea.l     stream,a0          Point at channel name
                                moveq     #0,d1              Permanent ownership
                                moveq     #1,d3              OPEN_IN
                                moveq     #1,d0              IO.OPEN trap key
                                trap       #2
                                tst.l     d0
                                bne.s     no_input
                                movea.l   d6,a1
                                move.l    a0,prefix+serial_id(a1)
link_it      movea.l   d6,a0              Restore A0, link address
                                moveq     #28,d0             Set up MT.LPOLL trap key
trapl_out   trap      #1
it_worked  moveq     #0,d0
                                rts
*
no_input    move.l    d6,a0              Deallocate the heap space
                                move.l    d0,d7
                                moveq     #25,d0            MT.RECHP trap key
                                trap       #1
                                move.l    d7,d0              Return OPEN error code
                                rts
*
find_pos2   bsr.s     find_pos           Extra call allows trapping
oops        rts
*
point_off   bsr.s     find_pos2          Is our interrupt linked?
                                bmi.s     no_error           If not, return at once
                                move.l    serial_id(a4),d0
                                beq.s     not_opened         No need to close it
                                movea.l   d0,a0
                                moveq     #2,d0              IO.CLOSE trap key
                                trap       #2
*
* Remove interrupt server and deallocate linkage memory
*
not_opened  lea.l     -prefix(a4),a0
                                moveq     #29,d0            MT.RPOLL trap key
                                trap       #1
                                moveq     #25,d0            MT.RECHP trap key
                                bra.s     trapl_out
*
* BUTTON%(x%) reads buttons and returns a code 0 to 7 (+)
*
button      movea.w    $112.w,a2          Vector gets words
                                jsr       (a2)
                                bne.s     bad_exit
                                subq.w    #1,d3              Ensure just one parameter
                                bne.s     bad_param           Reject otherwise
                                move.w    0(a1,a6.l),d3

```

Mouse Calls

Before you can use the mouse you must issue two commands - one to tell the

serial ports the baud rate and one to start the interrupt server. It is not necessary to open the serial port as the handler will do this automatically if it is not already open.

Normal QLs support only one RS-232 speed or 'baud rate' at a time - this should be 1200 baud for a PC serial mouse. If the wrong baud rate is selected the mouse will work erratically or not at

all. If you can't get the mouse to work, the baud rate is probably not set correctly. To get underway, use these commands:

```
BAUD 1200
PTR_ON
```

The Hermes upgrade from TF Services makes the serial ports much more reliable, among other tricks, and allows separate baud rates for each port. This means you can use a 9600 baud printer or modem at the same time as a 1200 baud serial mouse, but in such a case the BAUD command should not be used as that affects both ports. Set up your other port as normal, load the Hermes Toolkit, then enter:

```
t=RXBAUD%(196)
PTR_ON
```

to use a mouse with SER2, or:

```
t=RXBAUD%(68)
PTR_ON
```

to allow 1200 baud input from SER1 without affecting the other serial data streams.

Keywords

The keywords are the same as those for the Amiga Qdos pointer driver, so that programs that work with one will suit the other, although the Amiga mouse hardware is very different from that of a serial mouse.

You can read the co-ordinates from SuperBasic, set limits, and move the pointer to a desired integer co-ordinate. Amiga allows co-ordinates less than zero but the DIY Toolkit extensions only use positive co-ordinates, to a maximum of 32767.

The DIY Toolkit imple-

mentation adds a function to read the buttons. Amiga Qdos converts their signals into key-codes, so you cannot tell if more than one button is pressed. BUTTON% removes this restriction. It takes one parameter, like its namesake in SAM Basic; zero reads all the buttons, and other values

read buttons one, two or three (hardware permitting) independently.

If you have a three button mouse, BUTTON%(0) returns a value from zero to seven, with one bit corresponding to each button. For instance the value 5 is 101 in binary, indicating that the outer buttons are both

pressed and the middle button is not.

The binary digits tally with the buttons from left to right, with the tail of the mouse stretching away from the user, so button 3 is the left-most, indicated by the most significant bit. BUTTON%(0) returns a value of four, or more depending on the

```
*
* Put the state of all the buttons into D0; bit 0 = Button 1
*
read_all    bsr.s    find_pos
            move.w    button_bits(a4),d4 Check up to 16 buttons
            beq.s     ret_integer    None so parameter irrelevant
            tst.w     d3
            beq.s     ret_integer    Parameter 0, no filtering
            cmp.w     #3,d3
            bhi.s     bad_param
            subq.w    #1,d3          Number bits from 0
            btst      d3,d4
            bne.s     ret_integer    If choice set, return all
main_line    moveq    #0,d4
            bra.s     ret_integer    Stack space is allocated

*
* SYNCH needs just four lines and ten bytes of code
*
synch        bsr.s    find_pos
            moveq    #0,d4
            move.b    synchro(a4),d4
            bra.s     chk_integer

*
* XPOINT% and YPOINT% differ only in the offset they use
*
xpoint       moveq    #latest_x,d5
            bra.s     get_word
ypoint       moveq    #latest_y,d5
get_word      bsr.s    find_pos
            move.w    0(a4,d5.1),d4 Read co-ordinate variable

*
* Return the integer in D4.W to SuperBASIC via the RI stack
*
chk_integer  moveq    #2,d1          Number of bytes needed
            move.w    $11A.W,a2      Find the BV.CHRIX vector
            jsr       (a2)           Allocate RI space
            subq.l    #2,$58(a6)     Update BV.RIP
ret_integer  move.l    $58(a6),a1     Get BV.RIP
            move.w    d4,0(a1,a6.1) Stack the result
            moveq    #3,d4           Type = 16 bit Integer
no_error     moveq    #0,d0           No run-time error
            rts

*
* FIND_POS - point A4 at the parameters in the interrupt list
*
find_pos     moveq    #0,d0           MT.INF
            trap      #1
            move.l    #'DIY1',d7
            lea.l     60(a0),a4       Locate polled list
find_loop    move.l    (a4),d0
            beq.s     not_found       Pass error to prior caller
            movea.l   d0,a4
            cmp.l     marker(a4),d7 Check signature
            bne.s     find_loop
found_it     lea.l     prefix(a4),a4 Point at the data
            rts

*
not_found    addq.l    #4,a7           Discard one return address
            moveq    #-7,d0
            rts

*
bad_param    moveq    #-15,d0         BAD PARAMETER error
bad_exit     rts
```


other buttons, if the third button is pressed.

If you find the combinations confusing, it is easy to test the buttons individually. `BUTTON%(3)` would return zero unless the button was pressed; similarly `BUTTON%(1)` and `BUTTON%(2)` read the others, independently. In fact the code can cope with up to sixteen buttons, so it may be adapted for use with serial digitisers and other devices festooned with auxiliary switches. If you need more buttons you'll need to change the 'mask' applied to the first byte of each message from the mouse to let more bits through.

X and Y Limits

The two main functions are `PTR_X%` and `PTR_Y%`, which return the current position of the mouse, updated as it moves. `PTR_MAX` sets the limits for X and Y values - the default is `PTR_MAX 511,255` so that co-ordinates correspond to Mode4 pixels.

The Amiga has a four-parameter command, `PTR_LIMITS`, to set minima and maxima in each dimension. For compatibility with compiled tasks that use `PTR_LIMITS`, the DIY driver recognises the command, but the first two parameters are ignored and should be zero for compatibility. Only the last two parameters are used - the code just discards the first two, checks there's something left, and continues through `PTR_MAX`.

The co-ordinates are reset to zero by `PTR_OFF` followed by `PTR_ON`, so you may need a `PTR_POS` command after `PTR_ON` if you do not want the pointer to start in the corner. You can change the default by adding a line to set `LATEST_X` and `LATEST_Y` appropriately in the code after `POINT_ON` which initialises the other variables like `STEP` and `LIMIT_X`.

```

100 REMark Sinclair QL World HEX LOADER v 3
110 REMark by Marcus Jeffery & Simon N Goodwin
120 :
130 CLS: RESTORE : READ space: start=ALCHP(space)
140 PRINT "Loading Hex..." : HEX_LOAD start
150 INPUT "Save to file...";f$
160 SBYTES f$,start,byte : STOP
170 :
180 DEFine FuNction DECIMAL(x)
190 RETurn CODE(h$(x))-48-7*(h$(x)>"9")
200 END DEFine DECIMAL
210 :
220 DEFine PROCEDURE HEX_LOAD(start)
230 byte = 0 : checksum = 0
240 REPEAT load_hex_digits
250   READ h$
260   IF h$="*" : EXIT load_hex_digits
270   IF LEN(h$) MOD 2
280     PRINT"Odd number of hex digits in: ";h$
290     STOP
300   END IF
310   FOR b = 1 TO LEN(h$) STEP 2
320     hb = DECIMAL(b) : lb = DECIMAL(b+1)
330     IF hb<0 OR hb>15 OR lb<0 OR lb>15
340       PRINT"Illegal hex digit in: ";h$ : STOP
350     END IF
360     POKE start+byte,16*hb+lb
370     checksum = checksum + 16*hb + lb
380     byte = byte + 1
390   END FOR b
400 END REPEAT load_hex_digits
410 READ check
420 IF check <> checksum
430   PRINT "Checksum incorrect. Recheck data.":STOP
440 END IF
450 PRINT "Checksum correct, data entered at: ";start
460 END DEFine HEX_LOAD
470 :
580 REMark Space requirements for the machine code
590 DATA 650
600 :

```

`PTR_INC` sets the number of mouse pulses that correspond to one character-move in either direction. It is intended for the next version of the DIY Mouse driver, which will send cursor-control characters to tasks normally controlled from the keyboard. The increment is in Mode 4 pixels, and starts off at six in X and ten in Y, suitable for Mode4 text in `CSIZE 0,0` and `CSIZE 3,1`.

If you use Mode8 or wider characters the cursor will respond much faster to horizontal moves than to vertical ones, and it may be worth increasing the X increment, with a command

like `PTR_INC 12,10`. It pays to experiment - some people like slower vertical movement since it reduces the risk of moving off the current line when scanning left and right.

You can use `PTR_INC` to scale the movement on screen to suit your mousepad. Staying in Mode8 for the time being, try `PTR_INC 30,25` for easy character-positioning, and `PTR_INC 6,5` if you want faster movement. Double the first value, the X increment, if using Mode4, and double the second value to suit double-height characters.

So far `PTR_INC` is dor-

mant, as this version of the mouse driver does not use the `X_STEP` and `Y_STEP` variables. Notice that almost all the code it uses is shared with `PTR_POS`, `PTR_LIMITS` and `PTR_MAX`. The only difference is the offset of the stored parameters in the variable area.

Where am I?

The mouse driver does not display a pointer itself, as this could be over-written at any time by other screen output, and the code to intercept every access and update the pointer would be long and

tricky. In many cases this is unnecessary as the position of the cursor, bat or highlight indicates the current location on screen, and this can be updated by the main program.

The SuperBasic in Action Mouse Organ used the Amiga's hardware sprite pointer. QL SuperBasic programmers can use BLOCK, POINT, LINE or CURSOR with PRINT to position a marker on the screen, as I showed in last month's SuperBasic examples.

Note that vertical co-ordinate zero is the top of the screen, and moving the mouse down increases the co-ordinate. This suits BLOCK and PLOT, but is the opposite way round from the graphics co-ordinate scheme, which starts with line zero at the bottom of the window. To reverse the Y co-ordinate, use CHAN_W% from DIY Toolkit volume C to read the window height in pixels:

```
REPEAT dots : POINT
PTR_X%, CHAN_
W%(#1,30)-1-PTR_Y%
```

The POINT command does not report an error if asked to plot outside the window. The SCALE and PTR_MAX commands can make the mouse co-ordinate range an exact fit for your graphics window.

Buffering

I tested this code with a standard 128K QL, Gold Card, Sinclair JM Qdos, Hermes and Sinclair's version 0.7 IPC, and various Minervae. I concentrated on the tricky Sinclair hardware, as other Qdos machines have 'proper' serial ports with dedicated control chips which are much less likely to lose data.

There is a small delay between the mouse sending information and the co-ordinate variables being updated. In extreme cases the buffers may be filled and some messages may be ignored, but

this is unlikely to be a problem unless the machine's attention is diverted for a long time by an operation like LBYTES or FORMAT.

The mouse driver takes steps to re-synchronise itself if buffers overflow. Since last month I have found out more about the format of the two-button Microsoft mouse messages. It turns out that they have an eight bit resolution in X and Y, but the bits are split between three bytes. This means that the first two bits of each byte can be used to identify the position

in the message.

The first byte starts with two set bits, followed by a bit for each button, and two bits each to signal X and Y movement. These are the most-significant bits of the signed values in the next two bytes, which always start with one set and one reset bit. To find the full change in X, move bits 0 and 1 of the first byte into bits 6 and 7 of the second. Likewise bits 2 and 3 of the first byte slot into bits 6 and 7 of the third, Y byte, which are initially 1 and 0.

Last month's SuperBasic demonstration only used the six least significant bits. The new code uses all eight, so it can handle bigger changes in one message. Byte values are signed, so bytes over 127 correspond to their unsigned value minus 256. The code can re-sequence itself, rejecting bytes without the top bit set and treating bytes which start 11 as the first in a sequence.

At any time the QL's Intel 8049 second processor can store up to 23 characters en route from either serial port

```
610 DATA "43FA021834780110", "4ED2616267507220"
620 DATA "740070184E414A80", "665645FA012C214A"
630 DATA "000421470008217C", "01FF00FF0010217C"
640 DATA "0006000A00142C08", "70004E414AA8009C"
650 DATA "661641FA01CE7200", "760170014E424A80"
660 DATA "661022462348001C", "2046701C4E417000"
670 DATA "4E7520462E007019", "4E4120074E75617A"
680 DATA "4E7561FA6B70202C", "0010670620407002"
690 DATA "4E4241ECFFF4701D", "4E41701960CE3478"
700 DATA "01124E92667C5343", "66763631E800614A"
710 DATA "382C000C67364A43", "6732B67C00036260"
720 DATA "5343070466267800", "6022612E7800182C"
730 DATA "000E600C7A006002", "7A02611E38345800"
740 DATA "72023478011A4E92", "55AE0058226E0058"
750 DATA "3384E80078037000", "4E7570004E412E3C"
760 DATA "4449593149E8003C", "2014670E2840BEAC"
770 DATA "000866F449EC000C", "4E75588F70F94E75"
780 DATA "70F14E7561D4508C", "601047EB0010BBBC"
790 DATA "62EE61C6588C6002", "61C0347801124E92"
800 DATA "66E0554366DA3231", "E8006BD448413231"
810 DATA "E8026BCC28814E75", "202E009C67F82440"
820 DATA "327800E249EB0014", "4E9166EA74001401"
830 DATA "522C000E102C000E", "72C0B03C0002672C"
840 DATA "6A661942000FC401", "B401662A7030C02C"
850 DATA "000FEA4864040000", "00023940000C60C8"
860 DATA "197C0001000E1942", "000F60E0C202B23C"
870 DATA "0080670864EA422C", "000E60AC0202003F"
880 DATA "7203C22C000FED49", "82424881302C0000"
890 DATA "D0416B10322C0004", "B041630230013940"
900 DATA "00006084424060F6", "C202B23C00806704"
910 DATA "64AE60C20202003F", "720CC22C000FE949"
920 DATA "82424881302C0002", "D0416B10322C0006"
930 DATA "B041630230013940", "0002609A426C0002"
940 DATA "6094000653455232", "49520006FE560750"
950 DATA "54525F4F4646FDE4", "065054525F4F4E00"
960 DATA "FEEA0A5054525F4C", "494D49545300FEEA"
970 DATA "075054525F504F53", "FEDA075054525F4D"
980 DATA "4158FEC207505452", "5F494E4300000004"
990 DATA "FE64065054525F58", "2500FE5E06505452"
1000 DATA "5F592500FE1A0742", "5554544F4E25FE3C"
1010 DATA "0653594E43482500", "0000", "*", 50246
```

to the main processor; Qdos maintains a second queue for up to 80 bytes. The DIY code seems to track the mouse fairly well with a Sinclair IPC and JM Qdos, but button changes can get lost if the button is pressed while the mouse is moving fast.

Hermes expands the second processor buffer to 31 characters, and solves most problems of 'serial over-run' where the 8049 loses its place in the buffer. It works perfectly with this mouse driver, and is warmly recommended for serial mouse users.

The debugging function SYNCH% indicates which byte of the mouse message is due next. It starts with zero, and increments by one after each byte, so it gives two when a Y delta is expected. If bytes are corrupted or lost, SYNCH% is reset to zero, ready for a new and complete message. I've found SYNCH% useful in testing and it only needs ten bytes of code.

Assembler list

Listing One is the first half of the assembly code for the DIY Mouse. It includes routines to create and delete the mouse interrupt handler and variables. This is more complicated than the typical DIY Toolkit extensions, so it uses advanced assembler features like conditional assembly, introduced in the HPDUMP routines earlier this year.

You can configure the source for either SER1 or SER2, but the code must be reassembled to take account of the correct queue address. It is not enough to change the text in the file from "SER2IR" to "SER1IR" or vice versa.

Two equates at the start of Listing One control the port and mouse protocol. Set SER_PORT to one or two, and BUTTONS to two or three, depending to suit your mouse and cable, then re-

assemble the code to make a custom version. The DIY Toolkit disk volume includes all the variations, ready assembled.

The subsequent IFNE and ENDC directives ensure that an error occurs if the variables are set incorrectly. It's worth doing this if your assembler can cope, because it much reduces the risk of creating an invalid file. Devpac reports a 'user error' if called upon to assemble a FAIL directive; other assemblers may see it as an incorrect opcode. The comment after FAIL identifies the exact error.

DIY Mouse keeps ten values and an interrupt linkage in the common heap. The next set of EQUates give names for offsets in this area, making the source more readable and reliable. You can add more variables, for lower limits or extra details, by incrementing VAR_LENGTH and adding new offsets after SERIAL_ID. I shall discuss the variables in more detail next month.

PTR_ON and PTR_OFF come first. The serial channel is only closed by PTR_OFF if it was opened by PTR_ON. There is no code to set the baud rate as MT.BAUD might upset Hermes. It's better to use SuperBasic to set the exact speeds you want for each stream.

Romtable

This interrupt handler will run from rom, unlike the DIY Toolkit timer routines in Volume H. The mouse driver keeps its variables in common heap memory, and locates them when necessary by searching the interrupt list, looking for a characteristic 'signature' - in this case, "DIY1" - after each linkage.

This would slow down the timer routines, as the search must be performed when each extension is called, but I may change the timers to work that way if I produce a DIY Toolkit rom. For now you

can see the required code in the listing, and adapt it if you need to write your own romtable interrupt handler.

All SuperBasic extensions call the FIND_POS subroutine, which returns directly with a 'not found' error if the end of the list is reached and no signature is found. To prevent this error, turn on the interrupt handler with the PTR_ON command. Initially the pointer handler is loaded but not active; after PTR_ON you can turn it off at any time with the PTR_OFF command.

The interrupt handler takes very little CPU time, even if there are several bytes to be processed, but it is convenient to be able to remove it completely, releasing the serial port and discarding the variables in the common heap.

Repeated calls to PTR_ON or PTR_OFF do not generate an error. If the required serial port is not open when PTR_ON is executed, the code opens the channel, and it normally remains open till PTR_OFF closes it. If the channel should close unexpectedly the interrupt server notices and returns quickly without taking any action.

You are welcome to open the serial port before calling PTR_ON. In this case the server will 'steal' bytes from the serial input queue as they arrive.

It is much more efficient to read bytes directly from the queue with the IO.QOUT vector than it is to read them from the device with TRAP #3 calls. This is easy to arrange as two Qdos system variables point to the serial input queues whenever they are active. The long words at offsets 152 and 156 hold the addresses of the queues for SER1 and SER2 respectively, or zero when not in use.

The functions PTR_X%, PTR_Y% and BUTTON% read DIY Mouse variables and return their values to SuperBasic. Notice that BUTTON% skips the potentially

slow call to BV.CHRIX, returning an integer result in the space previously occupied by its parameter.

Next issue the listing will continue with the remaining SuperBasic extension routines, and source for the interrupt handlers that read serial bytes and adjust LAT_EST_X, LATEST_Y and BUTTON_BITS accordingly.

Multipointing

The QL has two serial ports but I only own one serial mouse, so I have not experimented with 'multipointing' - my buzzword for the use of more than one mouse at a time. The device name is configurable to SER1 or SER2, and you can run two interrupt servers at once, as long as the identifiers are different, but I suspect that even the mighty Hermes might have trouble coping with two asynchronous streams of mouse-droppings at once.

The pointer limits are global, rather than local to each task, so it is wise to stick to the defaults if you want to use several pointer-reading tasks at once. A clever task could monitor SV_KEYQ and reset PTR_MAX to suit itself whenever its cursor was re-selected.

The DIY serial mouse driver should run alongside the one built-in to the Amiga Qdos emulator, but the commands and co-ordinate functions would have to be renamed to avoid a clash with the Amiga system extensions; the device name should be "SER1IR" as standard Amiga systems have only one serial port.

I welcome reports from anyone who tries this, and I'm even more interested to know why they need to be able to point in four dimensions at once. I might try to develop a four-note version of the Amiga Mouse Organ, but I'm not sure I'd be able to play it once the code was running! As ever, I welcome your comments.

Beginners' Machine Code

Finally, in Part 6 Alan Bridewell begins to turn machine code to practical use.

One of the biggest problems that confronts would-be machine code programmers is this: it's all very well moving numbers around and getting them to do very simple jobs, but how do we do the more complicated - and usually more necessary - jobs? For example, almost any worthwhile program will need to open channels, write text to the screen and to the printer, and be able to accept data from the keyboard, regardless of what the ultimate purpose of the program is. And if you want a half-decent screen display you will also need to alter ink and paper colours, resize the characters, and draw interesting designs on the screen.

Text to window

Let us consider, for a few moments, just one of these problems in a more detail. Suppose we already have a suitable window opened (SuperBasic has three as soon as the QL is switched on, so it's a reasonable assumption), and we wish to produce a program which will write some text into the window.

We already know that each word in the screen ram is translated by the QL into a row of different coloured pixel dots (four dots in eight-colour mode, and eight dots in four-colour mode). So clearly a single character may need only part of a word, or more than

one word, in a row, depending on how wide the character will be. It will, of course have to occupy several rows of dots; how many, will depend on how high the character is. So working out how to print a single character of a required width and height at a particular location on the screen, with a particular ink and paper colour would appear to be a pretty formidable task - and we haven't considered flashing characters yet!

Fortunately, we have a major shortcut, and an obvious one if you think about it. As soon as you switch on your QL, you can press any character key on the keyboard, and, (surprise, surprise) that character appears in window #0 on the screen. This can mean only one thing: the code needed to solve the problem of printing characters on the screen already exists in the rom of the QL. All we

need to do is to use that code in our own programs.

Using QL code

If we can discover how to access the code used by SuperBasic, we can do anything in our machine code programs that can be done in SuperBasic. (Actually, we can do more than that.

There are routines in the QL rom which cannot be used directly from SuperBasic, but we can still use them in our machine code programs.) Only if you wish to do something exceptional do you have to produce your own complex code. For example, the authors of Lightning and Minerva have rewritten rom routines in order to improve them. Now, that is the job for

```
.....
OPEN A CON CHANNEL
.....
THE ROUTINE EXPECTS THE SUPERBASIC CALL TO PLACE IN:-
D1 BORDER COLOUR & WIDTH
D2 PAPER & INK COLOUR
D3 WINDOW WIDTH
D4 WINDOW HEIGHT
D5 X ORIGIN
D6 Y ORIGIN
:
:
:
MOVE.W D1,(A1)+ ; BORDER COLOUR/WIDTH IN CON TABLE
MOVE.W D2,(A1)+ ; PAPER & INK COLOUR IN CON TABLE
MOVE.W D3,(A1)+ ; WIDTH IN CON TABLE
MOVE.W D4,(A1)+ ; HEIGHT IN CON TABLE
MOVE.W D5,(A1)+ ; X ORIGIN IN CON TABLE
MOVE.W D6,(A1)+ ; Y ORIGIN IN CON TABLE
LEA.L CON,A1 ; RESTORE TABLE POINTER
MOVE.W $C6,A2 ; UT_CON VECTOR IN A2
JSR (A2) ; OPEN CONSOLE CHANNEL
CMPI.W #$00,D0 ; IS THERE AN ERROR ?
BEQ.S STORE ; IF NOT MISS OUT ERROR MESSAGE
; OTHERWISE WRITE ERROR MESSAGE
; IN #0 AND RETURN TO SUPERBASIC
; PROGRAM
MOVE.W $CA,A2 ; UT_ERR0 VECTOR IN A2
JSR (A2) ; WRITE ERROR MESSAGE IN #0
MOVEQ #$0,D0 ; NO ERROR RETURN
RTS
MOVE.L A0,(A1) ; STORE CHANNEL ID
:
:
.....
WRITE TEXT IN CON WINDOW
.....
:
MOVE.L (A1),A0 ; CHANNEL ID IN A0
LEA.L TEXT,A1 ; BASE OF TEXT IN A1
MOVE.W $D0,A2 ; UT_MTEXT VECTOR IN A2
JSR (A2) ; PRINT TEXT
RTS ; RETURN TO SUPERBASIC
:
:
SPACE RESERVED FOR CHANNEL PARAMETERS
:
DC.W $0000 ; PAPER & INK COLOUR
DC.W $0000 ; WIDTH
DC.W $0000 ; HEIGHT
DC.W $0000 ; X ORIGIN
DC.W $0000 ; Y ORIGIN
:
SPACE RESERVED FOR CHANNEL I.D.
:
TEXT TO BE WRITTEN IN THE WINDOW
:
DC.B "THIS IS A MESSAGE " ; CHARACTERS
:
.....
```

Listing two

```

100 z=RESPR(256)
110 LBYTES flip2_LISTING1_code,z
120 REPEAT loop
130 PRINT#0,"Open channel and write text? (Y/N) "
140 n = CODE(INKEY$(~1))
150 SELECT ON n
160 = 89,121: open_and_write:NEXT loop
170 = 78,110: STOP
180 = REMAINDER
190 END REPEAT loop
200 DEFINE PROCEDURE open_and_write
210 CLS
220 INPUT,"BORDER COLOUR? ";border_colour%
230 INPUT,"BORDER WIDTH? ";border_width%
240 INPUT,"PAPER COLOUR? ";paper%
250 INPUT,"INK COLOUR? ";ink%
260 INPUT,"WINDOW WIDTH? ";width%
270 INPUT,"WINDOW HEIGHT? ";height%
280 INPUT,"X ORIGIN? ";x%
290 INPUT,"Y ORIGIN? ";y%
300 CALL z,border_colour%*256+border_width%,paper%*256+ink%,width%,height%,x%,y%
310 END DEFINE

```

an expert!

Even jobs like writing other characters not in the standard set is not really a major task. If you wish to work out how to write to your screen in Arabic, Chinese, Cyrillic, Hebrew, Greek, or any other non-Roman characters, basically, you simply have to tell the QL to access your special font rather than the standard one in the rom. The biggest problem is designing all the character fonts, and then working out how to keep all the bits of sticky paper on the keys to tell you which key prints which character!

So, how do you find out about these routines in the rom? Various authors have published this information. Which source suits you is probably a matter of taste. You could try QL Technical Guide by Tony Tebby and David Karlin, QL Advanced User Guide by Adrian Dickens, or Advanced QL Machine Code by Adam Denning. Adrian and Adam probably got much of their information originally from Tony and David (Tony wrote Qdos in the first place!) However, Adrian and Adam have more in the way of examples and illustrations of how to use the code. (Much of what I know I learned from them.)

Two routine types

There are far too many

routines to explain in these articles. What I shall do is to go through a few of them to illustrate the different ways they can be accessed and used.

The routines may be divided broadly into two types, which require rather different handling. These are the "vectored utilities" and the "trap calls". For the moment we shall confine ourselves to vectored utilities.

These routines are held in the rom, but different versions of the rom have routines which may be of different lengths, and held at different addresses. So how can we write a program to use them which will automatically work whatever version of the QL we have?

All versions of the QL rom have a set of words from address \$C0 to address \$12B which contain the word addresses of the routines. This set of words is called the "vector table". What makes this so useful is that every version of the QL rom has the same location in the vector table for routines which do the same job. This means that all your program has to do is use the vector table to find the address of the routine you want, and then use that address. As long as your program uses the correct location in vector table, it will access the correct routine, regardless of where it is held in the rom.

This is a very important

point about using vectored routines, and must be understood if they are to be accessed correctly. Let us look at an example, which will be used later in this article.

There is a routine in the rom which allows us to open a CON_ channel. The

address of this routine is held at address \$C6 in the vector table. (The value \$C6 is known by the label UT_CON, and you may well find that your assembler will already accept the label UT_CON in place of \$C6.) This much is true of all versions of the QL. But the actual address of the routine itself, ie what we actually find at address \$C6, might be different in different QLs. However, as long as we use the address found at address \$C6, we will get the routine we want. And when people decide to improve on the routines, they can ensure their new roms are compatible with existing QLs by keeping each rewritten routine's new address at the same place in the vector table. Clever, isn't it?

Address registers

Since we know how to find the routine we want, there are various ways we might get at it. But the usual way of doing it is to put the address of the vector table location into an address register. For example

```
MOVE.W $C6,A2
```

will put the vector table address (which holds the address at which the "open a CON_" channel routine can be found) into register A2. We can then go straight to the routine with the line

JSR (A2)

The JSR stands for "jump to the subroutine". A2 means we use the address in register A2. However, the brackets means that the data found at that address is itself an address, and that is the address we actually jump to.

For these subroutines to work, they need to be given some data to start with, so they know exactly what to do. This is done by putting this data, in an appropriate form, into one or more registers.

CON_channel

To open a CON_ channel requires border colour and width, ink and paper colours, window width and height, and x and y origins (top left corner). This data must be placed in a "parameter table" in the correct order, which is held somewhere in ram. The starting address of this table must be held in register A1 before we jump to the routine. The exact format of this table is

byte	border colour
byte	border width
byte	paper/strip colour
byte	ink colour
word	window width
word	window height
word	x origin
word	y origin

If all has been successful, the newly opened window will clear in the given paper colour, with a border of the given colour and width. Also register A0 will contain the "channel ID" of the newly opened channel. This is rather like the channel numbers used in SuperBasic when channels are opened, used, and closed. But they are not the same numbers, and are not interchangeable. This channel ID is chosen by Qdos (unlike SuperBasic channel numbers, which are chosen by the programmer). It must be carefully stored, because it will be needed every time we want to do

anything with the channel, for example, resize or recolour, write text or draw lines, and of course close when we are finished with it.

Channel ID

Many routines expect to find the channel ID in register A0, so it can often be simply left where it is, as long as we can be certain it will not be overwritten before it needs to be used. You can also store it on the "stack", which is normally the clever way of doing things, but can lead to complications if you are using the stack for other things at the same time. The safest thing to do is to allocate a space for it within your program, and store it there. This is a bit more cumbersome, and not the way experts usually do it, but at least you know exactly where the thing is whenever you need to use it again.

But suppose something goes wrong. If, for example you give it an impossible size or position, the routine will be unable to open the channel. Does this mean your program will crash, which in the middle of a machine code routine often means the QL will hang up? Not if you handle things properly.

All routines in the rom where there is any possibility of things going wrong produce an "error return". This is a number which appears in register D0 on returning from the routine. If there is no error, the number zero will appear in D0. If an error does occur then a negative number will appear in D0, and the actual number will tell us what the error is. So by reading D0 after the routine is finished, we can find out if an error has occurred, and if it has, we can decide what action to take next, without the QL hanging up on us.

Error message

One very useful thing to

do is to write an error message in SuperBasic's channel #0 to tell us what went wrong. To do this we have a vectored routine called UT_ERR0 with an address in the vector table of \$CA. All this does is read the number in register D0, and print

an appropriate error message in #0. We can use this with, for example, the two lines

```
MOVE.W $CA,A2
JSR (A2)
```

Having opened a chan-

nel, we should to do something with it, and the most common thing is to write some text. For this there is a vectored routine called UT_MTEXT with an address in the vector table of \$D0. This routine expects to find the channel ID in register A0.

Listing three

```
100 REMark Sinclair QL World HEX LOADER v 3
110 REMark by Marcus Jeffery & Simon N Goodwin
120 :
130 CLS: RESTORE :READ space:start=RESPR(space)
140 PRINT "Loading Hex...":HEX_LOAD start
150 INPUT "Save to file...":f$
160 SBYTES f$,start,byte:STOP
170 :
180 DEFine FuNction DECIMAL(x)
190 RETurn CODE(h$(x))-48-7*(h$(x)>"9")
200 END DEFine DECIMAL
210 :
220 DEFine PROCedure HEX_LOAD(start)
230 byte=0:checksum=0
240 REPEAT load_hex_digits
250   READ h$
260   IF h$="" :EXIT load_hex_digits
270   IF LEN(h$) MOD 2
280     PRINT "Odd number of hex digits in: ";h$
290     STOP
300   END IF
310   FOR b=1 TO LEN(h$) STEP 2
320     hb=DECIMAL(b):lb=DECIMAL(b+1)
330     IF hb<0 OR hb>15 OR lb<0 OR lb>15
340       PRINT "Illegal hex digit in: ";h$:STOP
350     END IF
360     POKE start+byte,16*hb+lb
370     checksum=checksum+16*hb+lb
380     byte=byte+1
390   END FOR b
400 END REPEAT load_hex_digits
410 READ check
420 IF check<>checksum
430   PRINT "Checksum incorrect. Reread data. ":STOP
440 END IF
450 PRINT "Checksum correct. Data entered at: ";start
460 END DEFine HEX_LOAD
470 :
480 REMark Space requirements for the machine code
490 DATA 102
500 :
510 DATA "43FA0040": REMark          LEA.L      CON,A1
520 DATA "32C1": REMark             MOVE.W     D1,(A1)+
530 DATA "32C2": REMark             MOVE.W     D2,(A1)+
540 DATA "32C3": REMark             MOVE.W     D3,(A1)+
550 DATA "32C4": REMark             MOVE.W     D4,(A1)+
560 DATA "32C5": REMark             MOVE.W     D5,(A1)+
570 DATA "32C6": REMark             MOVE.W     D6,(A1)+
580 DATA "43FA0030": REMark          LEA.L      CON,A1
590 DATA "347800C6": REMark          MOVEA.W    $C6,A2
600 DATA "4E92": REMark             JSR         (A2)
610 DATA "0C400000": REMark          CMPI.W     #$00,D0
620 DATA "670A": REMark             BEQ.S       STORE
630 DATA "347800CA": REMark          MOVEA.W    $CA,A2
640 DATA "4E92": REMark             JSR         (A2)
650 DATA "7000": REMark             MOVEQ      #$00,D0

660 DATA "4E75": REMark             RTS
670 DATA "43FA0022": REMark          .STORE
680 DATA "2288": REMark             LEA.L      ID,A1
690 DATA "43FA001C": REMark          MOVE.L     A0,(A1)
700 DATA "2051": REMark             LEA.L      ID,A1
710 DATA "43FA001A": REMark          MOVEA.L    (A1),A0
720 DATA "347800D0": REMark          LEA.L      TEXT,A1
730 DATA "4E92": REMark             MOVEA.W    $D0,A2
740 DATA "4E75": REMark             JSR         (A2)
750 DATA "0000": REMark             RTS
760 DATA "0000": REMark             .CON
770 DATA "0000": REMark             DC.W       0000
780 DATA "0000": REMark             DC.W       0000
790 DATA "0000": REMark             DC.W       0000
800 DATA "0000": REMark             DC.W       0000
810 DATA "00000000": REMark          .ID
820 DATA "0012": REMark             DC.L       00000000
830 DATA "544849532049532041204D45535341474520": REMark DC.B    $12
840 DATA " ",7224                    DC.W       "THIS IS A MESSAGE "
```


(As I mentioned, if a routine is using a specific channel, it expects to find the channel ID in A0.) It also expects to find the address where the text to be written is stored in register A1. But this must be stored in a special format.

The first word contains the number of characters to be written. A mistake in this number will lead to some characters left off the end, or some gibberish tagged onto the end. The number is followed by the actual characters to be printed.

A program

That's enough new theory for now. Let's put it to work by producing a program to open a channel and print some text in it. In order to make it interactive (that is, so that we can play around with it from the keyboard while the program is running), we shall write it so that the channel parameters can be input from the keyboard in SuperBasic. You should recall that the SuperBasic CALL command will accept extra parameters which are put into registers before the routine is run. We can use this to put the channel parameters into the program.

Listing one shows our assembler program. We start by using the LEA instruction to put the address of the parameter table in register A1. By looking further down the listing, you can see the label CON is the first address of the space put aside for the channel parameters. At the moment it contains nothing but zeros. But if we use the CALL parameters correctly from SuperBasic, then registers D1 to D6 will contain the data for the channel parameter table. So we can now fill the parameter table from the registers with a series of post-incremented

MOVE instructions. This means that after each MOVE instruction, register A1 will contain the address for the next parameter.

Before we can use the UT_CON routine, we need to restore register A1 to point to the start of the parameter table with another LEA instruction. After this we can attempt to open the channel with the two lines mentioned earlier.

Successful?

We next want to know if we have been successful. By using the CMPI instruction we can find out if register D0 contains zero. If it does, the routine was successful, and we jump to two lines which store the new channel ID in a ram location put aside for it. (In this particular program, this step is not actually needed, because we are going to use the channel ID in register A1 before it has a chance of being corrupted. But in a longer program, where the channel ID might be needed many times, it needs to be stored safely, so we are doing it here for the sake of good practice.)

If D0 contains anything but zero, it means an error occurred and the channel was not opened. So the program goes straight on to the next two lines which print an error message in #0. Having printed the error message, we need to know what to do next. In a more complicated program we may wish to jump back to an earlier stage in the program to try to correct the error. But in this program, this cannot be done from within the machine code, so there is nothing left but to use RTS to return to SuperBasic. Before we do this, we put a zero in register D0 so that we do not get another error message on returning to SuperBasic. (Again, if you are following

this closely, you will realise that as the return to SuperBasic will print the error message for us, there is no need to use the UT_ERR0 routine. We could have simply left the error return in D0 and let SuperBasic print the error message for us. However, in general, we probably would not want to abort the machine code in this way, so it is worth while knowing about using the UT_ERR0 routine.)

Writing text

Having a channel in which to write text, we can now do just that. First we use the LEA instruction to fetch the address of the channel ID into register A1, and then use it to MOVE the channel ID into register A0 where it is needed. (In this program this is not needed, because the channel ID is already there, but in general we will normally need to put the channel ID back into A0 before we can continue.) Next, we load the address of the text into A1, before using the two lines needed to access the UT_MTEXT routine to print the text.

Finally, we return to SuperBasic with an RTS.

Listing two is a SuperBasic program to run the machine code. It assumes the machine code is in a file LISTING1_code in drive flp2_. You may need to alter this to suit your system. It then goes into a loop which first asks if you wish to open a channel and write text. If you don't the program will stop. If you do, it will request input of the channel parameters, and then use them as CALL parameters for the machine code. You will notice that parameters which are successive bytes have been combined to make words.

It is worth noting that the

program commits the sin of never closing channels, so if you use it enough times without resetting the QL, you will eventually run out of spaces for storing channel IDs. We have done this because closing channels requires a different technique, and we shall look at this next time.

Random channels

Having got the program to run successfully, you should try experimenting with it. You can do the obvious things, like changing the text to be printed, or perhaps set up the SuperBasic loop to continuously print in randomly generated channels. It is worth experimenting to see how error codes work. You can deliberately use impossible parameters and see what error message it gives. More interestingly, you can try deleting the RTS line which aborts the program when an error occurs, and, using impossible parameters so a channel does not open, and see what happens. Surprisingly, the program may still print the text on the screen.

There are other vectored utilities which can be quite useful for the beginner, although most do require a bit more expertise to get grips with. However to list them all together with exactly what they do, what parameters they need and where they are needed is far more than this article can run to. If you are really interested in using vectored routines then you ought to get one of the published books on machine code.

As before, listing three contains the code in Marcus and Simon's Hex Loader for those who do not have an assembler.

Happy coding!

Ultraprint 1.88

Bryan Davies tests a high-spec screen dump utility.

INFORMATION

Program: Ultraprint 1.88

Price: £19.95

Supplier: Digital Precision Ltd., 222 The Avenue, Chingford, London E4 9SE. Tel. 081 527-5493.

This program is a **screen-dump utility**. Commands for dumping the screen are built into some interface cards, and it might seem unnecessary to employ a separate program to do the job. Even so, printing what is displayed on the screen is not as straightforward as it should be. Some users will not wish to delve into descriptions of SuperBasic extensions to find the necessary command, anyway. If it does what is required of it, Ultraprint would be a worthwhile accessory for anyone wanting to dump screen images to a printer at anything more than odd intervals.

In my own experience, the three things the built in Trump Card and Gold Card SDUMP command does not do adequately are:

- i) print all four of the Mode4 colours, in the FX-80 mode,
- ii) allow easy adjustment of the printed image size, and
- iii) work regardless of whatever else is running.

The first point was a sore one for years, until experimentation with the SDUMP parameters yielded a solution. One of the four colours (green) would not print out every time a screen

dump was needed to illustrate a program review, it was necessary to fiddle with the RECOL command to try and get sections of screens to print out. The review program would often kill the effects of RECOL before a print could be made. The answer was to set the command to print to a different type of printer (Epson LQ-2500). This would not work for many users, who do not have a printer that can emulate this particular one. A problem with the RECOL command is that you have to be able to step out to SuperBasic to use it, and some programs (such as Quill) prevent that. Its effectiveness depends upon the program concerned not issuing any INK or PAPER commands to upset the RECOL settings subsequently.

Not green

What the basic reason for ignoring green is, I do not know, but Ultraprint missed it out the

same way as the interface dump command does, when run in Epson FX-80 mode. Setting the printer to LQ-2500 mode enables green areas to be printed, by both Ultraprint and the interface dump. In this respect, Ultraprint did not meet one of the three criteria stated above, but it may be that there is something fundamental to an FX-80 - or to FX-80 printer drivers - that prevents green being "seen".

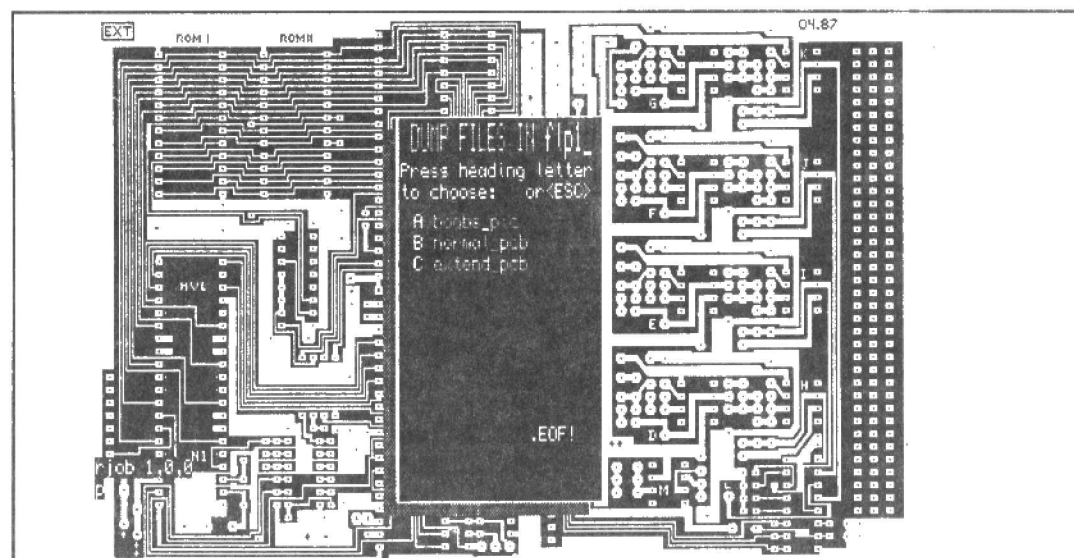
Figure one shows the screen with the file selection menu displayed. The third of the supplied images (identified as "C") is in the background of this illustration, behind the file-selection menu. This print was produced by the Gold Card interface SDUMP command, not by Ultraprint itself, since Ultraprint is not designed to capture "live" images of the screen. You do not have to go to SuperBasic to get a directory, as the program itself lists those files which it can print. Up to 10 files are listed in one window, with any remaining ones being

available through presses of the cursor keys.

Rather more functionality is available from the interface SDUMP command when the QJump Pointer Interface is used. It is possible with this to dump an area, or a window within that area, saved by the PSAVE command. It does not look possible to vary the size of the whole screen image, however. These features are not available on the basic QL, even with an interface card fitted, unless the Pointer Interface is loaded. In this respect, Ultraprint certainly scores, allowing the whole image to be re-sized. An important factor for many users is the comparative lack of study needed to be able to obtain satisfactory prints with Ultraprint.

Affirmative

Adding together the various points, there appears to be room for this separate screen



dump program.

Ultraprint is supplied on either cartridge or disk, whichever is requested. A working copy of the disk version can be made with a copying command, such as WCOPY. A clone program is provided for cartridge users to make a copy. One other difference between cartridge and disk versions is that the disk version has three screen dump routines on it - one for Epson printers, two for OKI printers (recent, and older, models) - whereas the cartridge version has only the Epson routine. This is because there is not enough space on cartridge to hold the OKI routines too, but you can order them instead of the Epson one. The routine name is SDUMP, the same as the Trump Card or Gold Card command.

The program can be multi-tasked (using Ctrl-C for switching) with other programs. It is Turbo-compiled, and requires some SuperBasic extensions, which are loaded automatically by the supplied boot routine. There are two extensions files, taking a total of just over 10 KB. The SDUMP routine itself takes about 23 KB.

Printing can be to either of the serial (the standard) ports, or to a parallel port. The latter will be available only on a few disk interfaces. Files to be printed can be taken from the devices mdv, flp, fdx and ram. An additional item on the device selection menu allows the free memory to be checked.

Paperwork support

The printed instructions cover 12 sides of A4 paper - a surprising amount for a single routine. In addition, there is an UPDATES.DOC file on the program disk, containing information not in the printed instructions. Fortunately, the updates with the review copy were brief - SDUMP now has a full-colour inversion facility, there are a couple of errors in a test program line, and OKI owners using microdrives are advised how to obtain the routine for older (eg 82A and 83A) models. The instructions are comprehensive and should be sufficient for almost all users.

There is a one-page section

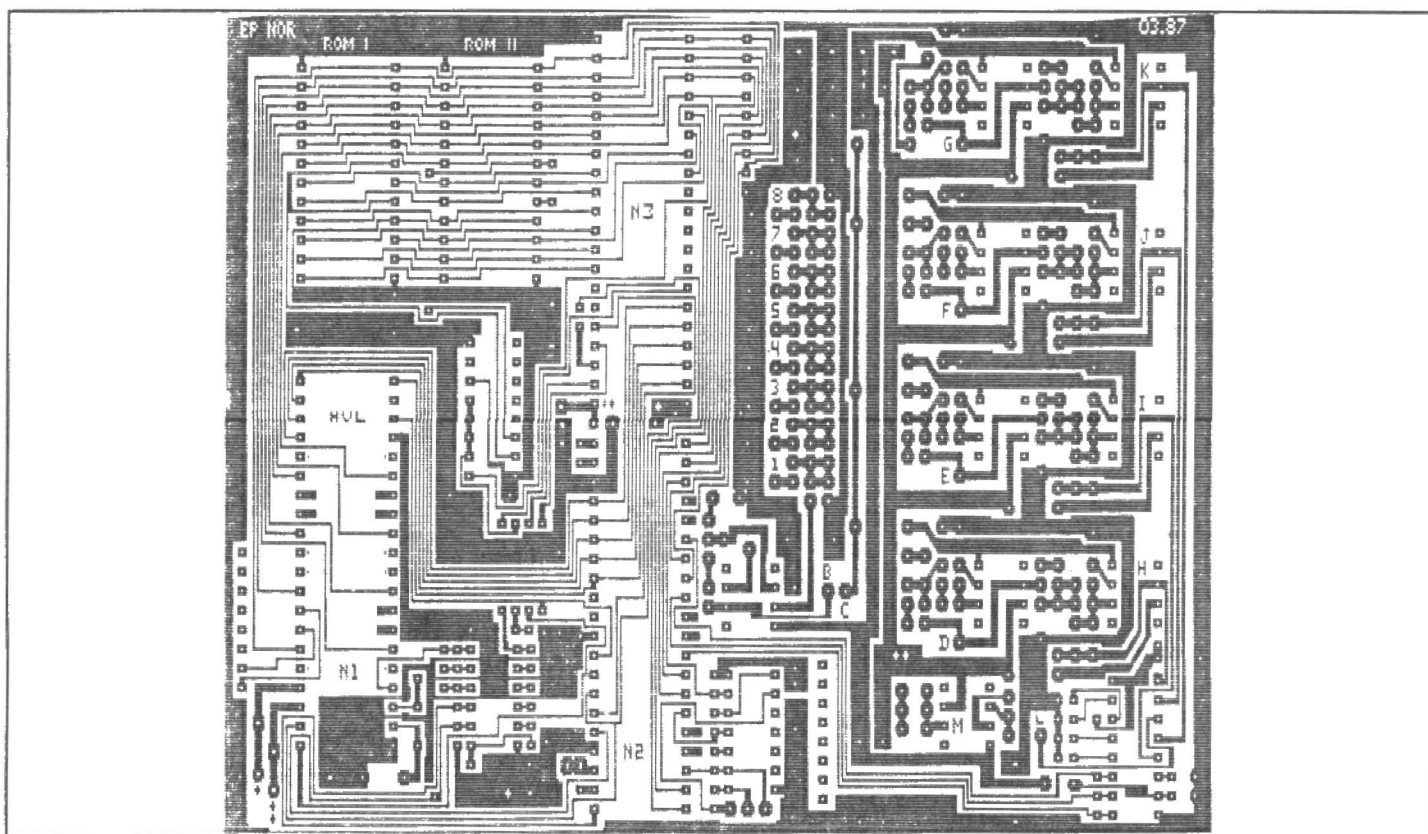
on multi-tasking, explaining how to run more than one copy of Ultraprint simultaneously with other programs. Printing is a slow operation at the best of times, and it is useful to be able to let Ultraprint get on with the job(s) while you do something else, such as writing a letter in your favourite WP program. It is pointed out that multi-tasking is no "free ride", though, and everything gets slower and slower as you run more and more jobs at one time. The Ultraprint screen may be confused when you switch back to it from another job but it can be refreshed by pressing any key.

Nearly two pages are occupied with advice on possible error messages from the program, and a further page and a half deal with trouble-shooting printer problems. A simple test is suggested for Epson compatibility - if your printer prints correctly from Easel with the standard, Epson printer driver, it is Epson-compatible. The last two pages of the instructions give SuperBasic listings for four tests, to determine the Epson-and/or OKI compatibility of printers. Likely users of Ultraprint include

people who have Eye-Q, and it is pointed out for these users that any file which is not exactly 32 KB in size will be loaded into Ultraprint by "compression loading"; if partial screens are saved to file from **Eye-Q**, the compressed mode should be used.

Check files

Three sample files are provided to check operation. Files which are to be printed must have one of the three suffixes _SCR, _PIC or _PCB. The user just rename existing files, to give them one of these suffixes, in order for SDUMP to find them. This makes the point that the program does not allow the user to dump the screen directly; screen images must be saved first. The instructions indicate that the dump is done from the memory image, not directly from the screen, but this statement might cause some confusion. If there is anything else on the screen, that may be printed as well as the stored image. To avoid unwanted material (such as the date and memory indicators which normally appear on my screen) turning up on the



printout, get rid of them before using Ultraprint.

The program can be used with printer types other than dot-matrix, but anything that does not print strip-by-strip may stop with an error message part-way through printing. For instance, my Epson GQ-5000 laser printer, working in FX-80 emulation mode, gave "buffer full" errors and took 2-3 runs to print single, minimum-sized images. This results in misalignment of the different sections of the image, as can be seen in **Figure two** of the "normal" printed circuit board layout. This print was done in inverse mode (compare it with Figure one). All the illustrations were actually done using the printer's LQ-2500 emulation mode. The reason for this was the one stated above to ensure coloured areas were not missed out - not because there was any improvement in quality.

Print modes

Two basic print modes are available, "normal" and "special". The Normal mode will dump eight colours (including black and white), representing them as shades of grey. There is a choice of three sizes, and images can be printed in negative (inverted) form if required that way. The function keys F1-

F5 are used to select the sizes, with Shift-F1-F5 giving the equivalent negative prints. The size increases with the function key number. An additional option, obtained with Ctrl-F1, gave greater contrast with the F1 size. The extra carriage width of Epson 100-series models can be utilised with the larger size, although a small section of the right-hand side of an image printed this way was "off the paper".

The Special mode is especially for subjects requiring high contrast, such as the two printed circuit board samples supplied. There are three normal size options, obtained from F1-F3, with corresponding Shift keyings for inverse- colour prints. To make use of wide-carriage printers, there are Ctrl-F1-F3 extended size options, and their Ctrl-Shift-F1-F3 counterparts for inverse colours. With the correct choice of size option, PCB prints can be "camera-ready". It is suggested that Ultraprint be used in conjunction with DP's Eye-Q graphics program, for CAD (computer-aided design) work. One of the sample pcb files was printed full-page A4, and the print was good, but I lost count of the number of times the paper had to go back into the printer to get the job done, and misalignment inevitably

occurred with each re-insertion. This problem would not occur with a dmp printer, but the quality would not be as good either.

Line spacing

There is a line spacing option, which allows corrections to be made for errors in shape, such as flattened circles. The thin white horizontal lines that spoil many dmp prints can be diminished in size, or almost completely removed, by suitable adjustment of the spacing. The increments used are 1/216-inch for Epson and 1/144-inch for OKI. There is also the usual option you get with DTP programs, to print in 1 to 5 passes; this should be necessary only if the printer ribbon is producing faint images.

A small change in the line spacing setting makes a big difference in the height of the print. The file from which **Figure three** was printed came from amongst those supplied with **Professional Publisher**. The printed clock faces changed proportion considerably with only a single-digit change in the setting; in this case, the setting was reduced from 5 to 4. The Special mode was used, and gave a much better print than the Normal mode.

The width of the printed image

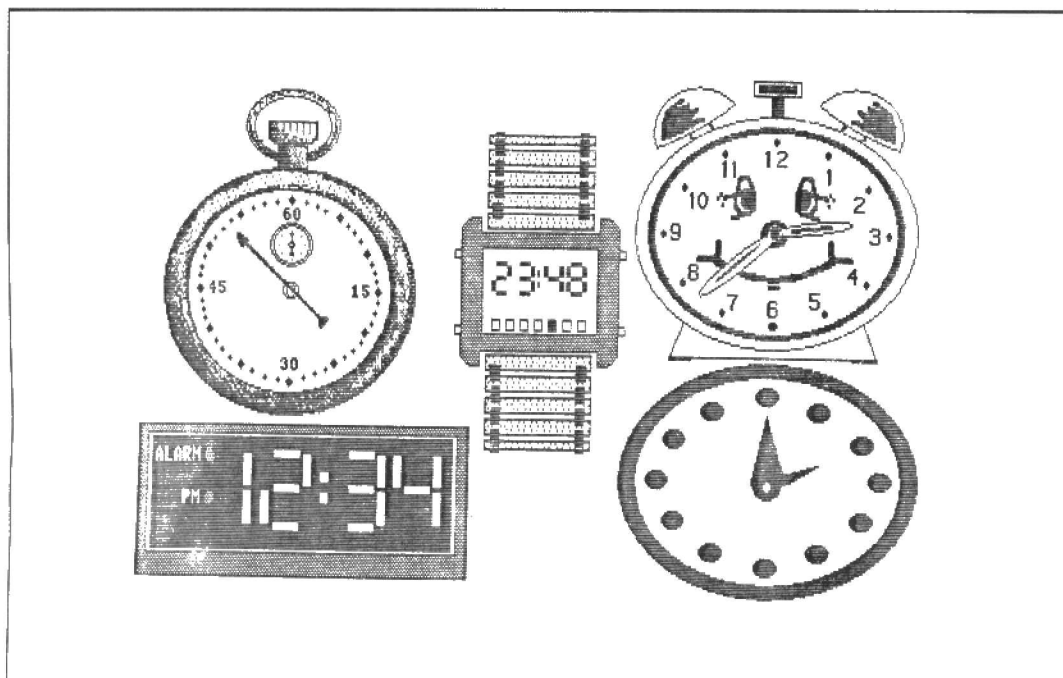
can be varied from about 10 cm to three times that. All test prints from the laser printer came out in landscape orientation. Quite a lot of prints were started part-way down the paper, such that they could not be completed before the edge of the paper. This may have been a function of printing on the laser printer, as the second "slice" was usually printed in the correct position. It is assumed this would not happen with a dot-matrix printer.

ED problem

There can be a problem for users of ED disks, as the program rejected two of them, initially with the message "device error". Reading the instructions, it was clear that the reason for this was that the write-protect tab was set on the disks, and the program wants to write something to any disk it accesses (not a desirable practice, as far as the protection of valuable files is concerned). Once the disks had been unprotected, the program halted, indicating that the Dataspace needed to be increased, possibly because insufficient has been allowed for the information in an ED disk directory. Not all users have the Turbo compiler and can adjust Dataspace, and this weakness might rule out the use of ED disks for some people. The required image files had to be transferred to a DD disk. This is unfortunate, since users of, for instance, Professional Publisher, are just the type of people who may use ED drives, as they are eminently suitable for holding the large volume of files associated with DTP work.

Conclusion

Ultraprint is not designed to give prints superior to those you can get using the interface SDUMP command, but it does provide greater flexibility in size and proportion of the printed image. It also allows "off-line" printing of saved screen images. It is not priced at the rock-bottom end of the market, but is within range of many users.



DJC

Dilwyn Jones Computing

41 BRO EMRYS, TAL-Y-BONT, BANGOR,
GWYNEDD, LL57 3YT, GREAT BRITAIN

FAX/TEL: (0248) 354023

DATABASE SOFTWARE

COCKTAILS WAITER £10.00	
DATA DESIGN 3	£60.00
DATA DESIGN API	£20.00
DBEASY	£15.00
DBPROGS	£15.00
FLASHBACK	£25.00
FLASHBACK SE	£40.00
SUPER DISK INDEXER	£12.00

DTP & CLIPART

PAGE DESIGNER 3	£40.00
Available at last!	
UPGRADE FROM PD2	£25.00
SCANNED CLIPART 1 (U)	£10.00
SCANNED CLIPART 2 (U)	£10.00
THE CLIPART (U)	£12.00

FILE HANDLING

FILEMASTER	£12.00
FILES 2 (U)	£12.00
4MATTER & LOCKSMITHE	£23.50
LOCKSMITHE (U)	£14.95
MDV TOOLCHEST	£14.95
THE GOPHER (U)	£12.00
WINBACK 2	£25.00

FILE TRANSFER

CONVERT-PCX	£10.00
DISCOVER	£20.00
MULTI-DISCOVER	£30.00
OPD INTERCHANGE	£15.00
QL-PC FILESERVER	£24.50
STOQL	£24.50
TEXTIDY	£15.00

GAMES & LEISURE

FIVE GAME PACK (U)	£12.50
FLEET TACTICAL COMMAND (U)	£39.95
FTC DATA PRINTER	£9.95
FLIGHTDECK (U)	£15.00
FUGITIVE	£9.95
GREYWOLF	£12.50
OPEN GOLF	£12.50
QUESTION MASTER (U)	£10.00
QUIZ MASTER 2 (U)	£12.50
RETURN TO EDEN	£17.50
SOLITUDE (U)	£15.00
SQUIDGY ROUND THE WORLD (U)	£12.50

LABELLING

ADDRESS BOOK & LABEL PRINTER	£15.00
SUPER DISK LABELLER	£10.00

GENEALOGY

GENEALOGIST 3	£60.00
new pointer driven version	
Upgrade 2nd Edition	£33.00
upgrade other versions, ask	
GENEALOGIST 2ND EDITION	£30.00
BUDGET 128K GENEALOGIST (U)	£12.00

GRAPHICS

IMAGE-D (U)	£10.00
IMAGE PROCESSOR	£15.00
LINE DESIGN	£100.00
PAINTER	£25.00
PICTUREMASTER	£15.00
PICTUREMASTER PLUS	£20.00
QRACTAL	£20.00
QUICK MANDELBROT 3 (U)	£15.00
SCREEN COMPRESSION	£10.00
SCREEN SNATCHER (U)	£10.00
SIMPLE VIDEO TITLES	£5.00
TRANS24 (U)	£10.00

HARDWARE

MICRO PROCESS CONTROLLER	
new redesigned version	£64.50
MPC SOFTWARE	£9.95
NETWORK PROVER	£4.00
QPOWER REGULATOR	£24.95
(unsuitable Gold Card)	
SERMOUSE	£40.00
VOICE ANALYSER-New!	
Ask for details and price	
Add £2.50 postage for MPC,	
SERMOUSE AND VOICE ANALYSER	

MAGAZINES

QREVIEW	£2.00
QL TECHNICAL REVIEW	£2.00
Issues 1-8 available	
QL ADVENTURER'S FORUM	£1.75
Issues 1-9 available	
QL LEISURE REVIEW	£2.00
Issues 1-2 available	
(UK prices only shown)	

PROGRAMMING

BASIC REPORTER (U)	£10.00
DISA	£29.00
DJTOOLKIT (U)	£10.00
EASYPTR 3 PART 1	£40.50
EASYPTR 3 PART 2	£20.00
EASYPTR 3 PART 3	£20.00
MEGATOOLKIT DISK (U)	£25.00
MEGATOOLKIT EPROM (U)	£40.00
QLIBERATOR 3 36	£50.00
BUDGET QLBERATOR (U)	£25.00
QLOAD AND QREF (U)	£15.00
S EDIT EDITOR	£20.00

SCREEN DUMPS

SIDEWINDER PLUS	£24.95
-----------------	--------

SPREADSHEET PRINTING

SIDEWRITER (U)	£15.00
3D TERRAIN	£12.50

SUNDRIES

3 5" DSDD DISKS each	£0.40
3 5" DSHD DISKS each	£0.70
3 5" DISK LABELS (roll 100)	£2.00
3 5" LABELS (printer roll)	£2.50
ADDRESS LABELS (roll 100)	£2.00
MICRODRIVE CARTRIDGES	£2.50
MDV LABELS (roll 100)	£2.00
MOUSE MAT	£2.50
3 5" DISK DIVIDERS (20)	£3.00

Add £0.50 postage for labels or mouse
mat if only ordering those, or £2.50 postage
for disks or disk box divider sets

TEXT

BANTER BANNERS	£25.00
BIBLE TEXT	£20.00
QTYPE	£29.95
QUICK POSTERS (U)	£10.00
ROB ROY PACK (U)	£10.00
SPELLBOUND	£30.00
SPELLBOUND S E	£50.00
TEXTIDY	£15.00
TEXT 'N' GRAPHIX	£20.00

OTHER SOFTWARE

HOME BUDGET (U)	£20.00
PRINTERMASTER (U)	£20.00
QPAC1	£19.95
QPAC2	£39.95
THE Pointer Environment package!	
QTOP	£29.95
SCREEN DAZZLER	£15.00
SCREEN ECONOMISER (U)	£10.00
SLOWGOLD (U)	£5.00
SPEEDSCREEN (U)	£15.00
SPEEDSCREEN EPROM (U)	£30.00
TASKMASTER	£25.00
VISION MIXER 1	£10.00
VISION MIXER PLUS	£22.50

CALL OR WRITE FOR A FREE COPY OF
OUR QL SOFTWARE CATALOGUE
DETAILING AROUND 100 QL PRODUCTS

PLEASE NOTE: (U) ABOVE MEANS THAT
THE SOFTWARE IS SUITABLE FOR USE
ON A 128K QL



TERMS: POSTAGE-software sent post free to UK, overseas add £1.00 per program (maximum £3.00) Floppy disks, SERmouse, etc add £2.50 postage (see above) PAYMENT - IN UK currency (pounds sterling) only, please. Valid methods of payment are cheque drawn on UK branch of bank or building society, Eurocheque, Postal Order, cash (send by registered post) or by credit card - Visa, Access, Mastercard and Eurocard accepted. Please make cheques etc payable to DILWYN JONES COMPUTING. Minimum order value £5.00 (due to bank charges). Goods remain property of Dilwyn Jones Computing until paid for in full. Orders normally sent out within a few days of receipt, we will attempt to advise if any delay anticipated due to stock problems. Orders can be accepted by telephone if paid for with credit card. Orders paid with credit card can only be sent to cardholder's address under card company rules.

FAX - We now have a Fax machine on our usual number (0248) 354023

